Deanship of Graduate Studies

Al-Quds University

Scholar Search Relevancy Optimization

Rida Rohi Taher Abdelmajid

M.SC Thesis

Jerusalem –Palestine

1441 / 2020

Scholar Search Relevancy Optimization

Prepared by:

Rida Rohi Taher Abdelmajid

M.SC: Al-Quds University Palestine

Supervisor: Dr. Badie Sartawi

A thesis submitted in partial fulfillment of the requirements for the degree of master of computer science at Al-Quds University

1441 / 2020

Al-Quds University

Deanship of Graduate Studies

Department of Computer Science

Thesis Approval

Scholar Search Relevancy Optimization

Student Name: Rida Rohi Taher Abdelmajid

Registration No: 21710524

Supervisor: Dr. Badie Sartawi

Master Thesis submitted and accepted, Date: 10/08/2020

The names and signatures of the examining committee members are as follows:

1-Head of Committee: Dr. Badie Sartawi Signature . [illegible]

2- Internal Examiner: Dr. Rashid Jayousi Signature ……[illegible].

3- External Examiner: Dr. Ahmad Ewais Signature ……[illegible]……..

Jerusalem/Palestine

1441 / 2020

Dedication

To my parents for their love and support throughout my life, to my dear wife, who has encouraged me all the way, to anyone appreciates knowledge, to all of them I dedicate this thesis work.

Rida Abdelmajid

Declaration

I certify that this thesis submitted for the degree of master is the result of my own research, except where otherwise acknowledged, and that this thesis (or any part of the same) has not been submitted for a higher degree to any other university or institution.

Signed.

Rida Rohi Taher Abdelmajid

Date: 10 / 08 / 2020

Acknowledgements

I would like to thank my supervisor Dr. Badie Sartawi for all the guidance and patience throughout this work. I would also like to thank him for the challenges he put in front of me, as I truly believe they have contributed to a better final work.

I would like to acknowledge and thank Dr. Radwan Qasrawi and Dr. Rashid Jayousi, for their excitement and willingness to provide feedback that made the completion of this research an enjoyable experience.

Abstract

The massive amount of information published every day has made it difficult to conduct a more relevant search for educational materials, because of loose requirements regarding metadata content. Many researchers have pointed out the importance of metadata and the efficiency of machine learning, deep learning algorithms, and language processing in text classification and semantic similarity by discussing the efficiency of algorithms without indicating how to use these algorithms to enhance search results based on metadata.

This study aims to test some of the natural language processing tasks, machine learning, and deep learning techniques, in addition to the role of metadata in enhancing search results. This study propose a framework called "Al-Quds System" as a recommender and assistance system to enables the author to add as much relevant information as possible to their publications to enhance retrievability by increasing the opportunity of displaying documents and texts in the search results.

This study shows the possibility of enhancing search results by combining language processing, deep learning, and metadata. In addition, this shows that documents can classified with high accuracy whether using traditional classifiers such as Multinomial Naïve Bayes (MNB) with 85% accuracy or deep learning such as Convolutional Neural Network (CNN) with 79% accuracy, and the accuracy of classifiers depends primarily on features extracting. In addition, the semantic similarity of words or sentences can computed by representing words into vectors, the closer the cosine value to 1, the smaller the angle, and the higher match between words vectors. Which had a positive impact on the Al-Quds system as a recommender and assistance system to use by the authors.

تحسين ملاءَمة نتائج البحث

إعداد: رضا روحي طاهر عبد المجيد

إشراف: د. بديع السرطاوي

ملخص:

مع الزيادة الهائلة في عدد المؤلفات والمنشورات التي يتم نشرها يومياً تزداد الحاجة الى اظهار النتائج ذات العلاقة بعمليات البحث. حيث أشار العديد من الباحثين الى أهمية البيانات الوصفية وكفاءة خوارزميات التعلم الالي، التعلم العميق ومعالجة اللغة في عملية تصنيف النصوص والوثائق والتشابه الدلالي للكلمات، وبالرغم من ذلك فان معظم الأبحاث والدراسات القائمة بهذا المجال ركزت على كفاءة الخوارزميات دون الإشارة إلى آلية استخدام هذه الخوارزميات بتحسين نتائج البحث بالاعتماد على البيانات الوصفية.

تهدف هذه الدراسة الى اختبار العديد من مهام معالجة اللغة، تقنيات التعلم الالي والتعلم العميق بالإضافة الى دور البيانات الوصفية وتوظيفها بمراحل تصنيف النصوص والوثائق من خلال تقديم إطار عمل يسمى نظام القدس للتوصية والمساعدة" بهدف تصنيف الوثائق والنصوص وتمكين المؤلفين من إضافة أكبر قدر من المعلومات ذات الصلة بأبحاثهم بغرض تعزيز نتائج البحث وزيادة فرص اظهار مؤلفاتهم ضمن نتائج البحث.

بينت هذه الدراسة أنه يمكن تحسين نتائج البحث واسترجاع النتائج المرجوة من خلال الدمج بين معالجة اللغة والتعلم العميق من جهة وبين البيانات الوصفية من جهة أخرى. كما بينت الدراسة أنه يمكن تصنيف الوثائق والنصوص بدقة عالية سواء باستخدام المصنفات التقليدية مثل Naïve Bayes بدقة 85% والتعلم العميق مثل CNN بدقة 79%. وأن دقة وجودة المصنف تعتمد في المقام الأول على عملية استخراج الميزات التي سيستخدمها، وانه يمكن حساب التشابه الدلالي بين

الكلمات أو الجمل من خلال تمثيل الكلمات على شكل متجهات وحساب زاوية جيب التمام، التي كلما اقتربت من 1، كلما كانت الزاوية أصغر وزاد التطابق بين المتجهات، وهذا كان له الأثر الإيجابي بالوصول الى نظام القدس كنظام مساعد فعال يمكن استخدامه من قبل المفهرسين والمؤلفين.

Table of Contents

List of Tables

List of Figures

List of Appendices:

List of Abbreviations

1. SEO: search engine optimization
2. NLP : Natural language processing
3. CNN: convolutional neural network
4. ANN: Artificial Neural Networks
5. RNN: Recurrent Neural Networks
6. CBOW: Common Bag Of Words
7. ML: Machin Learning
8. DC: Dublin core
9. NNLM: Neural net language models
10. MD: Metadata
11. SVM: support vector Machine
12. LRMI: Learning Resource Metadata Initiative
13. OAI-PMH: Open Archives Initiative Protocol for Metadata Harvesting
14. TFIDF: Term Frequency Inverse Document Frequency
15. DAN: Deep averaging network
16. NNLM: neural network language modeling
17. XML: Extensible Markup Language
18. RDF: stands for Resource Description Framework
19. TF-IDF: term frequency and inverse document frequency
20. The System: Al-Quds Recommended and Assistance system

Chapter1

Introduction

1.1 Introduction

The enormous increase in documents and information published every day makes it difficult to retrieve relevant documents and information. This limitation appears, despite the improvement made in the search engines and natural language processing domains. The accuracy of search results was measured by the relevancy of query terms with publication ranked displayed by search engines (Manral and Hossain, 2012), (Beel, et al, 2010). This relevance can be improved by many automated text and document classification algorithms, such as semantic text classification, rule-based, and statistical methods, the success of these machine learning algorithms relies on their capacity to understand feature extraction and dimensionality reduction within data (Kowsari, et al, 2019). Text classification contains different level of scope that can be applied:

1. Document level
2. Sub document level :such as paragraph or sentences
3. Metadata level

Classification based on metadata provide a common scheme across different formats, and can be used as a placeholder for the publication data in the classifier (Weber, et al, 2019). This in turn enhances the classifier performance and accuracy by reducing the amount of unstructured data (Kowsari, et al, 2019), to classify and find the most relevant keywords in order to optimize search results, especially title, abstract, and description of the publications or text. These attributes are mainly used by the search engine (Beel, et al, 2010), to predict the discipline of text or documents.

The general objective of this study is to exploit different natural language processing (NLP), and machine learning (ML) techniques involved in the search results optimization including the classification of publications, metadata, word similarity, and finally the ranking functions. The objectives will be done, by proposing Al-Quds System framework, which demonstrates a practical example of transfer learning of ML to different NLP tasks, by using an open source platform to build a machine learning model Tensorflow (TensorFlow, n.d.), (Weber, et al, 2019) with a deep learning library for python Keras (Keras, n.d.). Al-Quds System contains several models such as, the text classification model that informs authors and indexers in which categories their publication is. The sentences similarity check model, which computes the similarity and relevancy between sentences or words by encoding text into high dimensional vectors that are trained with a deep averaging network (DAN) encoder (Cer, et al, 2018). This model is an open source platform that builds machine learning model Tensorflow. Metadata model enables the authors and indexers to describe their works through metadata. Finally, the ranking model gives a notion for authors and indexers about their research results ranking.

The 20 newsgroup dataset (Home Page for 20 Newsgroups Data Set, n.d.), used in the experiment. It’s one of the most popular datasets used for machine learning, and contains approximately 20,000 newsgroup documents, partitioned across 20 different group.

1.2 Problems Statement

As previously mentioned, the massive amount of published information every day makes it difficult to display the most relevant search results, especially with the absence of the standardized classification vocabulary model (Kushwaha, 2015), metadata. Having repository in place contain tons of publications records is not enough. Another issue that relates to the way an institutional repository fits into its environment is that it can be isolated, as a single entity within the scope of an organization, or it cannot be aware of and interoperate with other systems to share, retrieve, or provide information. In addition, using a keyword for searching is flexible but the problem is the search result may not be relevant to the search query. Searching by controlled vocabulary, requires the author or cataloger to be familiar with these vocabularies which is considered less flexible than searching by keywords. Another challenge appears when using a search engine, in which every search engine uses their own ranking algorithms, some of its parameters known and others hidden. For instance, even search engine like Google probably return confusing results if publications metadata or description do not prepare correctly. While results appear according to the google ranking and evaluation algorithms, whereas adding relevant information about publication that considered culture, vary from authors or indexer to another. For even the author of the article may not agree with reviewing many publication, article, and conference. All keywords are given with a number indicating a calculated correlation that the key phrase is applicable to this document and depending on the academic background of these authors and editors (De Cock, et al, 2005).

Many approaches have been proposed by the literature to enhance the search results relevancy, especially in the text classification domain. These approaches can be broadly classified into three main categories: Classification based on metadata, classification based on text, and mixed approach by using both text and metadata (Richter and MacFarlane, 2005). However, these approaches have not been sufficiently discussed the impact of metadata on search results relevancy. Moreover, these approaches mostly discussed metadata extraction from the publications, without discussing methods that enable the author to add as much relevant information as possible to their publications using metadata.

1.3 Objectives of the study

1. Propose Al-Quds System as a recommender and assistance system to:

    1.1 Enables authors and indexers to classify their works.

    1.2 Enables author to add as much relevant information as possible to their publications through metadata.

    1.3 Check the relevancy of publication subject and abstract with descriptive keywords.

2. Exploit the capabilities of modern natural language processing, deep learning, and word embedding, for text classification and similarity check to:

    2.1 Investigate whether CNN can be efficiently used for text classification as it is mainly used for image classification by comparing CNN with Multinomial Naïve Bayes classifier.

    2.2 Comparing the classical text classification methods with deep learning methods.

3. Utilize metadata capabilities to optimize search results, by extend the Dublin core attributes by suggesting multiple description keywords for the publication.
4. Test hypothesis and contribute in the domain of optimizing the search result.

1.4 Motivation

The initial motivation presented in this thesis lies in learning how search engines and digital libraries can categorize a vast amount of information and return results quickly without cheating the ranking algorithms. In addition, develop of the Recommended Assistance System by utilizes different text categorization techniques to enhance the search results through enables the author to add as much relevant information as possible to their publications, reduce the possibility of a mismatch between the user’s actual information need and the interpretation of query keywords by a search engine, as describe in chapter 3.

1.5 Thesis Contributions

This thesis exposes optimization issues related to enhance search results. The research contributions can be listed as follow

1. Propose "Al-Quds system" framework as a recommender and assistance system to enhance the research results by:

1.1 Enabling authors and indexers to choose the most relevant word for publication subject, based on word embedding of the title abstract and description of their publications.

1.2 Enhancing metadata content by adding correlating and synonymous words in subject and description to increase the opportunities of search engines to displays relevant results.

1.3 Enabling authors and indexers to know the suggestion classification class of their publication by using classical or deep learning classifiers.

1.6 Hypothesis and Research Questions

Defining a framework that can be used for helping authors to prepare their publications to enable quick and relevant search results between search terms and display results is a challenging task. This needs a set of elements that must be defined accurately. The main assumption here is that enhancing scholar search results primarily depends on the publication preparation. This is done by using effective text classification and using relevant keywords through the metadata. Therefore, the main research questions that this research work tried to answer are:

Q1. How can search results be improved without overburdening the authors or researchers?

To answer this question, the content level will be taken into consideration. In which authors have the ability to define, and add relevant information with the help of Al-Quds system. This is done in order to contextualize the publication into related domains, using metadata, and synonyms keywords, which will enable quick and relevant retrieval results for the researchers. The System allows indexers and authors to examine the relevancy of abstract and subject with probable categories for their publications, adding relevant keywords and information in the metadata.

Q2.What effects do different SEO, and metadata parameters have on the relevancy of scholar search results?

This question will be answered by discussing the characteristics of some of the SEO, and metadata parameters. How one could take advantage of metadata, and many NLP tasks such as text classification and similarity check to define appropriate characteristics for some of these parameters.

1.7 Research Methodology

During this thesis work, and after reviewing the most updated and related contributions that have been proposed by the research community in the fields of metadata, machine and deep learning for text classification, search optimization, natural language processing (NLP). The mixed approach between quantitative and qualitative used, and examine through seven phases to fully understand the basic ideas behind search optimization and the NLP tasks, such as text classification, similarity checks, and sentiment analysis.

Phase 1: Comparing different related methods and algorithms, preparing a comprehensive review and discussion of the relevant scientific literature of metadata, text classification, and similarity check. Identify the challenges and gaps in the current literature related to search result optimization problems by implement and compare different techniques to tackle them, figuring out how to take advantage of the previous and current work in this domain to benefit it in the right way.

Phase 2: Formulating hypotheses and produce generalizable knowledge a wide and deep review of these topics.

Phase 3: Using a statistical test to test and analyze hypotheses and observations.

Phase 4: Testing the impact of features and hyper-parameters modifications on classifiers performance and accuracy.

Phase 5: Comparing the traditional and neural network-based classification methods, which builds and combine various expression to together including word embedding, pass it to optimizer which automatically finds out the gradient train the model, setup optimizer to predict output, and finally compute the loss and accuracy to update the model to work better by batch normalization and others parameters, where graphs and matrixes used to express and evaluate the results.

Phase 6: Using the transfer-learning concept on the Al-Quds system by using the result of pre-trained word vectorization to compute the distance between word vectors, to represent the similarity ratio, then use (DAN) with universal sentence encoder to find the relations between title, subject, and abstract.

Phase 7: Validating results by test the relevancy by using scoring and ranking functions such as (MB25), cosine similarity, and TFIDF.

1.8 Research Obstacles

The variety of topics covered by this study and the way addressed by researchers produced some obstacles such as:

1. Many techniques need to combine to create the final picture of an effective approach to enables faster and more accurate scholar search results. Where different areas to search. The domain that was discussed contains different areas, including metadata, NLP tasks ,machine learning, search engine optimization, and scoring functions
2. Time management: learning (python) to customize and build model consumes a lot of time.
3. Lack of resources: doing experiment for text classification need enormous data.
4. During my research and study process, the amount of display information from search was a challenge, especially when these results were not relevant to a search query, which requires more time and effort to find the desired information.

1.9 Thesis Structure

The remaining part of this thesis is structure as following:

- Chapter 2 (Background and literature review): this chapter summarize relevant literature review about metadata and different NLP tasks such as text classification, word and sentences similarity checks and ranking functions to handle scholar search optimization problem .
- Chapter 3 (Theoretical framework): this chapter consist theories and concepts and their definitions related to scholar search optimization problem

- Chapter 4 (Experimental Results and Discussion): this chapter describes the experiments setup, design, implantation and evaluation.
- Chapter 5 (Conclusion and Future works): This chapter summarize the main study results, and suggestion for future works.
- List of References and Appendices:

Chapter 2

Background and Literature Review

2.1 Background knowledge

Introducing the background material relevant to this study, especially the key terms of metadata such as metadata harvesting, OAI-PMH, OAI –PMH verbs, controlled vocabulary, to represents the basic knowledge upon which this study results are based on.

- **Metadata**

Metadata is defined as "structured information that describes, defines, locates, or otherwise makes it easier to retrieve, use or manage an information resource" (Barker and Campbell, 2010). The concept metadata has been used by libraries which provides an early example of metadata being applied in the physical model for classification and categorizing materials based on the title, author, and subject, to become more easily located on shelves from among many thousands of others books, the primary purpose of metadata remains the same, which is to allow locate the resources that the researchers looking for efficiently. In addition, metadata enhances resource discovery, identifying, organizing, this helps researchers to utilize digital resources efficiently, where the needs of the interoperability and integrating between these resources arise, which in turn raises a need to differentiate objects or resources, supporting validation and protection.

Metadata is a key in ensuring the data can easily be interpreted, analyzed, and processed by the data's originator. It enable data sets designed for a single purpose to be reused for other purposes, and over the longer-term metadata is an investment that may not be optional. Deng and Reese mentioned that the metadata interoperability does not depend on developing a set of standards on top of existing ones, but rather should extend the existing metadata schemas, which will be defined in the next chapter where the word embedding use to enrich the keyword that authors or indexer fill in the metadata in order to increase the opportunities to return the desired search results (Deng and Reese, 2009).

To enhance the search result, it's important to understand the added value of metadata roles and their capabilities, where metadata are the crucial elements, it can play a crucial role in collecting, managing and classification of publications, qualified metadata information is important for the authors, librarians, and searchers, where the amount of data increase and becomes more difficult to find specific information, but the main problem remains with a huge amount of information that usually lacks a good structure, that makes it difficult to manage.

Metadata is recognized as one of the key methods used for organizing such kind of digital data. The Dublin Core as one of the popular metadata stander (DCMI: Dublin Core^{TM} Metadata Element Set, n.d.) provides a small set of fifteen vocabularies intended to be used as metadata. Using these vocabularies help in classifying web content into common categories across the web. Dublin Core standard was intentionally designed to provide a basic set of elements that is both easy to apply, especially by non- specialist, and is suitable for a wide range of resource description communities, although an important focus appears to be on its usage in web documents, it can use in many contexts, as it is readable by both machines and humans because of its main characteristics as the flexibility and extendibility. Chapter four demonstrate a practical example of using DC .

- **Metadata Functions**

Metadata functions makes publications content more usable by identifying, managing, retrieving, and tracking usage of content, without these functions, the information will be scattered. The main metadata functions can summarize by:

- Content discovery and identifications.
- Organizing content
- Facilitating interoperability
- Archiving and preservation to ensure that resources will survive and continue to be accessible into the future.

- **Metadata Types**

A metadata standard will usually support a number of defined functions such as resources discovery and access by using a set of elements, (Riley, 2017) describes three types of metadata according to its functions :

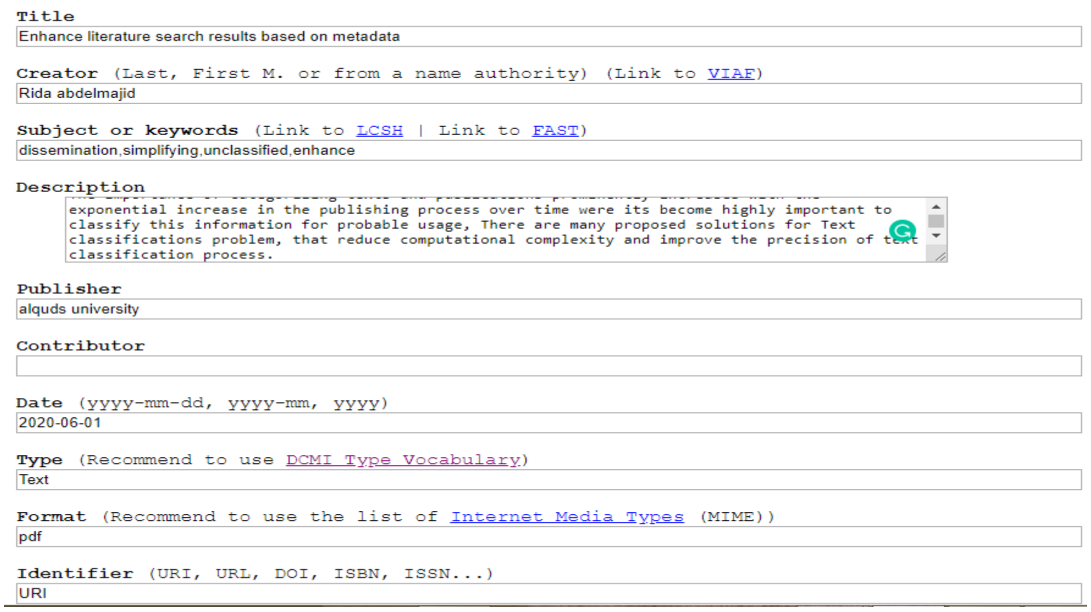
1. Descriptive metadata: use for finding or understanding a resource, often including the use of controlled vocabularies for classification and indexing and links to related resources.
2. Administrative metadata: used to decode and render files, Long-term management of files, Intellectual property rights attached to content.

3. Structural metadata: used to describe the structure, relationships of parts of resources to one another.

This study focuses in descriptive metadata type, it is essential for discovering publications, and includes necessary information such as title, abstract, subject, author.

To avoid the interoperability and integrity issues, this study proposes to extend the Dublin Core as standard of data structure, by adding additional description for the publication subject and categories, this extension does not conflict with OAI-PMH. Table (2.2) show an example of Dublin Core, Table (2.2) show the result of converting DC from HTML format to XML format.

Table 2.1: Dublin core attributes template



Title
Enhance literature search results based on metadata

Creator (Last, First M. or from a name authority) (Link to VIAF)
Rida abdelmajid

Subject or keywords (Link to LCSH | Link to FAST)
dissemination,simplifying,unclassified,enhance

Description
exponential increase in the publishing process over time were its become highly important to classify this information for probable usage, There are many proposed solutions for Text classifications problem, that reduce computational complexity and improve the precision of text classification process.

Publisher
alquds university

Contributor

Date (yyyy-mm-dd, yyyy-mm, yyyy)
2020-06-01

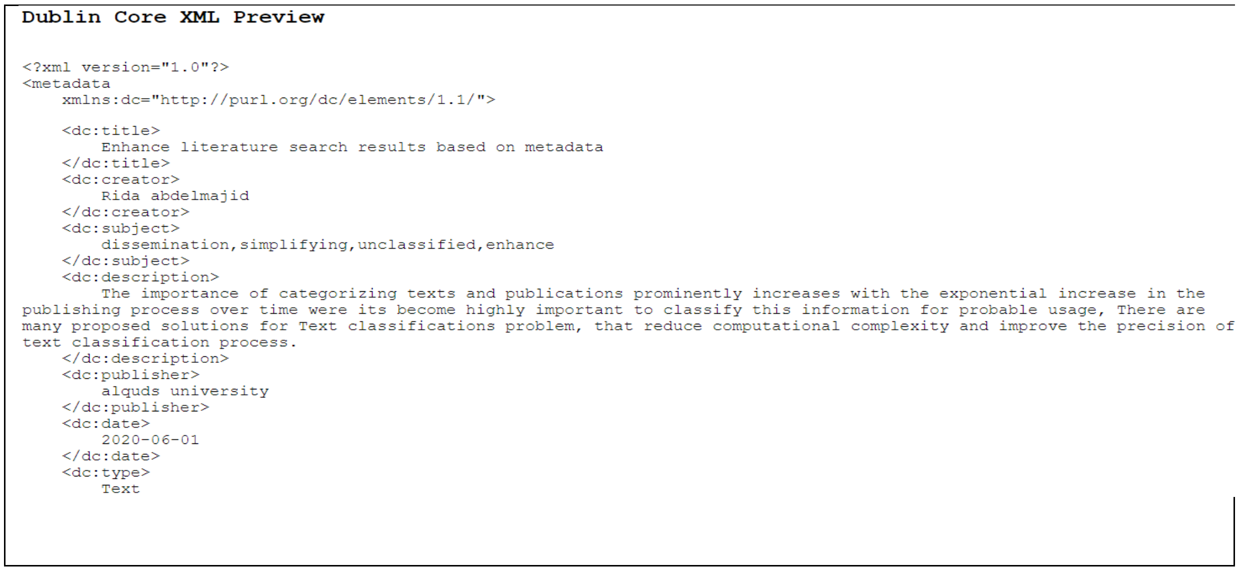
Type (Recommend to use DCMI Type Vocabulary)
Text

Format (Recommend to use the list of Internet Media Types (MIME))
pdf

Identifier (URI, URL, DOI, ISBN, ISSN...)
URI

Source : (DC - Template n.d.)

Table 2.2: DC XML preview



Dublin Core XML Preview

```
<?xml version="1.0"?>
<metadata
    xmlns:dc="http://purl.org/dc/elements/1.1/">

    <dc:title>
        Enhance literature search results based on metadata
    </dc:title>
    <dc:creator>
        Rida abdelmajid
    </dc:creator>
    <dc:subject>
        dissemination,simplifying,unclassified,enhance
    </dc:subject>
    <dc:description>
        The importance of categorizing texts and publications prominently increases with the exponential increase in the
publishing process over time were its become highly important to classify this information for probable usage, There are
many proposed solutions for Text classifications problem, that reduce computational complexity and improve the precision of
text classification process.
    </dc:description>
    <dc:publisher>
        alquds university
    </dc:publisher>
    <dc:date>
        2020-06-01
    </dc:date>
    <dc:type>
        Text
```

- **Dublin Core**

Dublin Core (DC) is a metadata standard used to describe a wide range of network resources, it consists of fifteen elements (DCMI, n.d.), Table (2.3) shows fifteen elements of DC into three categories according to its functions, which addresses the most basic functions required to uniquely identify a digital resource, its semantics established through consensus of an international (ISO - ISO 15836-1, 2017), the Dublin Core elements set is a basic standard which can easily understood and implement.

The reason for choosing Dublin Core in this work refers to a set of characteristics, such as, the simplicity of creation and its semantics that are commonly understood, this allows a non-specialist to create a simple descriptive records for information resources, while providing for effective retrieval of those resources, the flexibility, the fifteen element consider as optional to fill by authors ,extensibility the DC metadata itself is encoded into XML, in addition, most of resource discovery metadata standards can map to the Dublin Core set

Table 2.3: Dublin Core Elements

Content	Intellectual Property	Instantiation
Coverage	Contributor	Date
Title	Creator	Format
Description	Publisher	Identifier
Type	Rights	Language
Relation		
Source		
Subject		

The scope of this study is the content elements category, because of:

1. The description term, may include but is not limited to an abstract, a table of contents, a graphical representation, or a free-text account of the resource.
2. The subject term typically, the subject will be represented using keywords, key phrases, or classification codes.
3. The title term, typically a title will be a name by which the resource is formally known.

- **Metadata Harvesting**

Metadata harvesting is the process of collecting metadata from distributed repertories into the combined data store (Awasthi and Jaiswal, 2008), it's the process where the data harvester collects metadata from the data provider (Roy, et al, 2017), harvester need to comply with (OAI-PMH) protocol

- **OAI-PMH**

The Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH), is a protocol used for metadata harvesting (Jayakanth, et al, 2005) to gather metadata from electronic repositories, the second release of this protocol based on client server architectures, its use XML over HTTP, the harvester represents the client that send http request to the repositories which represent the server, which responses with XML metadata format, it consist of a set of six verbs or services that are invoked within HTTP. Fig. (2.1) shows interaction of static repositories, gateway, and (OAI-PMH) harvester.

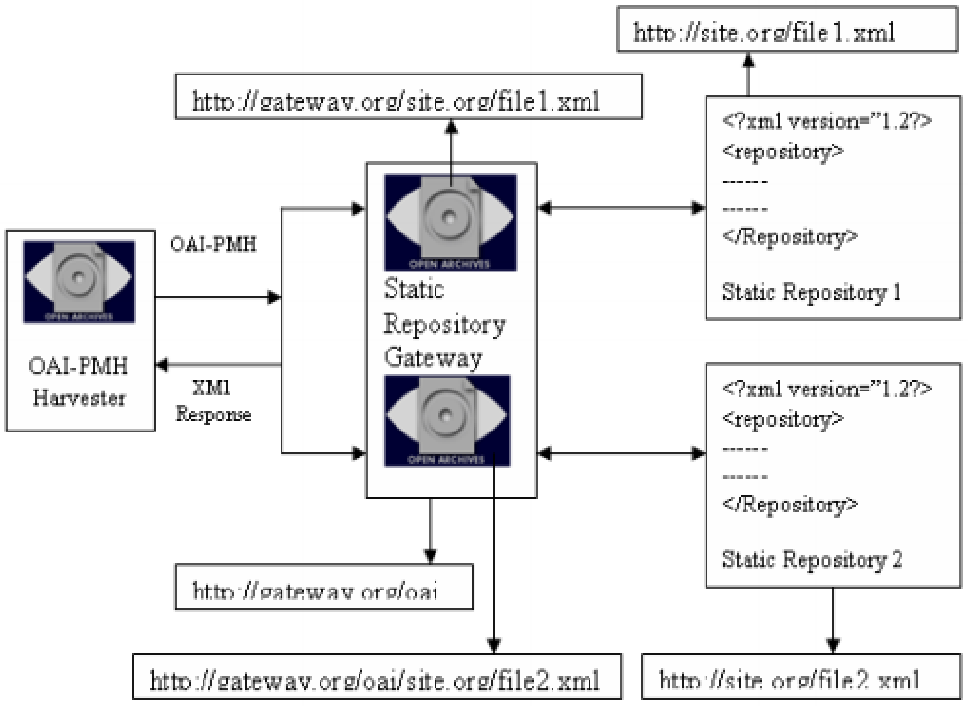


Figure 2.1: OAI-PMH Harvester
Source (Jayakanth, et al, 2005)

- **OAI-PMH Verbs**

The http protocol use six command to invoke metadata from repositories, these commands are called verbs (Awasthi and Jaiswal, 2008). Table (2.4) summarize the six verbs.

Table 2.4: OAI-PMH Verbs

OAI-PMH Verbs		
List Sets	Get Record	List Identifiers
List Records	Identify	List Metadata Formats

- **Control Vocabulary**

Control vocabulary is a predetermined list of terms on certain topic, which identifies preferred words or phrases for a given concept. It is often defined as a hierarchical relationships between terms. By using a control vocabulary, it could be enhance the text classification by using similarity checks or string matching (Golub, et al, 2007).

2.2 Literature Review

Enhancing scholar search results has been discussed widely and addressed with variant topics and domains, where many researches addressed the importance and roles of metadata and how it could serve in classification problems. Other researchers discussed the efficacy and accuracy of different machine learning and deep learning algorithms to solve classification problem. While other scholars discussed the methods used for search engine optimization, to increase the ranking score of publication. However, a few of researches discussed the idea of using all these information and contributions together in order to enhance relevancy between display search results with search query.

To provide a comprehensive look for achievements on these domains, this study will start from metadata, metadata harvesting, text classification methods, similarity check, natural language processing, word embedding, and finally reviewing search engine optimization, to take an overview of the current knowledge and identify the relevant theories, methods, and gaps in existing work.

The importance of metadata in the science of publication describes how metadata is used, take into consideration the creation, validation, standardization, and property rights (Pennington, 2014). Many research efforts have been invested in metadata extracting from publication with different extensions such as (PDF, HTML,X ML HTML) well discovered in the researches (Azimjonov, 2018), (Han, et al, 2003), (Marinai 2009) discussed extracting different fields of metadata from publication text using different methods show application that classified document to scientific and un scientific based on set of metadata attributes, they found the extracting result of metadata differs between metadata extractor, were it depends on the attributes that the extractor used such as title, abstract, and text. The impact of metadata on accuracy of automated classification significantly improve the classification process (Richter and MacFarlane, 2005), (Denecke, 2009), Metadata can be extract from different formats using different methods and techniques based on artificial intelligence, learnt statistical measurements, machine-aided indexing or automated indexing, heuristics, and machine learning. (Waltman and Van Eck, 2012), and (Akritidis and Bozanis, 2013) discussed the publication classification or taxonomy using machine learning tools and algorithms based on metadata.

(Richter and MacFarlane, 2005) show that the use of metadata could significantly improve the classification of patents with one classification system, improving classification accuracy from 70.8 up to 75.4 percent, and show that the result depends on the dataset the author used.

(Waltman and Van Eck, 2012) introduce a new methodology for constructing classification systems at the level of individual publications were publications are clustered into research areas based on citation relations, each publication is assigned to a single research area, and research areas are organized in a hierarchical structure at the highest level, research areas may, for instance, correspond with a broad scientific disciplines, at the lowest level, they may correspond with a small subfields, (Waltinger, 2018) discussed automatically classifying scientific documents according to the Dewey Decimal Classification (DDC) taxonomy within three levels using a machine learning-based classifiers SVM to classify OAI metadata records where addresses the problem of automatic enhancement and normalization of OAI metadata records in terms of subject indexing according to the DDC for document representation, only the content of the Dublin core such as title, subject, and description fields were used, the hierarchical machine learning classifier showed with an average F1-Measured of 0.61-0.81 promising result, (Weber, et al, 2019) discussed the possibilities to apply machine-learning algorithms on bibliographic data labeled with DDC numbers, another approach was to use SVM models to predict DDC research disciplines is presented which shows that assigning disciplines of research to metadata is a multi-label classification problem, research data can be mapped to multiple disciplines of research and these disciplines are not exclusive. Different approaches are used for auto text classification, such as rule base system, and machine learning based systems including deep learning and the hybrid systems, The rule-based has its limitations and, the machine learning approach can achieve superior result (Chiticariu, et al, 2013), (Kowsari, et al. 2019) especially when the text represents efficiently by using techniques such as bag of words (BOW), the most common methods used for text representation, where each word acts as a feature, this method has its own limitation especially that the word will lose its semantic (Zhang, et al, 2015). Other classification features which take into consideration the sequence of words by taking more than words or characters used such as N-gram which allows to represent a number of word as key phrases instead of a distinct words, this approach keeps the semantics and the syntactic of words (Náther, 2005), (Violos, et al, 2018) representation of words as continuous vectors has a long history (Mikolov, et al, 2013), (Pennington, et al, 2014), other studies show that using a simple model of neural network within single hidden layers is used to learn word vector representation

(Jang, 2019). Many research introduced techniques that can be used for learning high-quality word vectors representations from huge datasets (Sarkar, et al, 2015), (Mikolov, et al, 2013).

The machine learning (ML) approaches many methods and algorithms that are tested such as rule based systems, automated classifiers using the K-nearest neighbors algorithm, Naive Bayes classifier, and the support vector machine (SVM) which is widely used in supervised text classification (Lilleberg, et al, 2015). It could offer better performance as compared to other well know ML techniques (Sarkar, et al, 2015), The SVM is the nearest approach for text vectorization, where it depends on the state of art of converting text to vectors where SVM algorithm determine the best decision boundary between vectors that belong to specific class and which vector doesn't its simply draw the best line that divided the vector space into subspaces. (Akritidis and Bozanis, 2013) introduced a supervised machine learning algorithm for classifying research articles, and algorithm which operates by using a predefined list of labels based on three vectors that which correlates each article keyword, author, and journal the paper shows that inclusion of the aforementioned parameters leads to improved classification performance by roughly 6%.

A text classification approach was introduced that relies only on bibliographic metadata comprising title, conference name, journal-title, author's information, publisher and corporate creator (Jayakanth et al. 2005). (Awasthi and Jaiswal 2008) came up with a very preliminary classification of metadata items by enables the author to add as much relevant information as possible to their work to enhance retrievability without specifying which metadata items were needed to be added and how could be measured the performance and quality of the work.

(Pennington, et al 2014), (Kennedy and Ogada, 2016) describe an N-gram-based approach to text categorization that is tolerant of textual errors, The system is based on calculating and comparing profiles of N-gram frequencies. The N-gram frequency method provides an inexpensive and highly effective way of classifying documents (Kerstin and Denecke, 2009) combines rule based and machine learning technologies to classify catalogue entries based on N-grams and supervised ML classifier. A probabilistic algorithms like Naive Bayes and character level N-gram are some of the most effective methods in text classification (Violos, et al, 2018), The author proposed a text stream classification model where the text is represented as N-gram graph, the classification process takes place using text pre-processing, graph similarity and feature classification techniques following the supervised machine learning approach, in which the result shows that the performance score of Bayes classifier is better than SVM and Deep learning.

The recent research focusing on the text vectorization or on the other word converts text into vectors. Many research such as (Kim, 2014) discusses the idea of using convolutional neural network in text classification by experimenting the CNN trained on top of pre-trained word vector that keeps the semantic of words by producing semantic vector space model. It represent each word with rea value vector to reduce the computational power needed to compute the word or sentences embedding. Deep learning methods involve in learning word vector representations can be used as a features in variety of natural language processing model (Pennington, et al, 2014), the CNN model achieve excellent results in different NLP tasks, continuing of using the state of art word vectorization google researches in (Cer, et al, 2018) present model for encoding sentences into vectors to be use in different NLP they implement their model in Tensorflow that available to download from TF Hub (TensorFlow Hub, 2018).

Other works explore the performance of word2vec Convolutional Neural Networks (CNNs) to classify news articles and tweets into related and unrelated ones. Using two-word embedding algorithms of word2vec, Continuous bag of word (CBOW) and Skip-gram, to construct CNN with the CBOW model and CNN with the Skip-gram model, The experimental results indicated that word2vec significantly improved the accuracy of the classification model, The accuracy of the CBOW model was higher and more stable when compared to that of the Skip-gram model (Pennington,et al 2014), (Jang, et al, 2019). The CNN is just a feature-extraction architecture, alone itself is not useful, it is the first building block of a larger network. It needs to be trained together with a classification layer in order to produce some useful results representing the text in one dimensional array. Skip-gram model is an efficient method for learning high-quality distributed vector representations that captures a large number of precise syntactic and semantic word relationships describe here an N-gram-based approach to text categorization that is tolerant of textual errors, The system is based on calculating and comparing profiles of N-gram frequencies, the authors found that The N-gram frequency method provides an inexpensive and highly effective way of classifying documents (Kim, 2014), (Náther, 2005).

The advantages of using distributed representations of words in vector space for the learning algorithms in order to achieve better result were discussed in (Mikolov, et al, 2013), where they found that using the Skip-gram model and CBOW model increases the quality of the learned words and phrase representations. The sentences representations rather that characters or word representations discussed in (Cer, et al, 2018), in which they found an interesting approach used to propose a model for sentences embedding instead of word or character embedding. It introduces the model architecture for our two encoding models the transformer based encoder target height accuracy tat consume resource, the deep averaging network (DAN) targets efficient results with slightly decreased accuracy.

Web optimization and scholarly search engine optimization based on some of the SEO parameters are used in ranking algorithms such as google ranking algorithms, and while some variables of these ranking algorithm are known, others are hidden. Each search engine uses different ranking algorithms and different crawl methods that mainly index a keywords listed on abstract, title and descriptions where theses element need to be considered for optimization process, especially when work with publications which are usually stored in databases (Manral and Hossain, 2012), using more SEO parameters in ranking algorithm will help in retrieving better search result (Beel, et al, 2010).

2.3 Conclusion

Most of the metadata reviewed and researched focuses on the performance of machine learning algorithms rather than the advantages and usage of metadata, on the other hand, the research related to machine learning and deep learning mainly focuses on a techniques that can be used to optimize the classifiers by using several technique such as feature extractions and dimensionality reduction of words in vector space without discussing the aspects related to improving search results. Search engine optimizations methods discussed the search engine ranking functions optimizations without mentioning the need for text and publications classification and methods that can use to enhance the search relevancy such as similarity check. Based on review text classification summarize into two ways, manually and automatically each method has its own pros and cons, the scope of this review includes the automatic text classification method, grouped into three different types:

1. Rule Base system.
2. Machine Learning based systems.
3. Hybrid systems.

The rule-based systems is human comprehensible and can be improved but this approach has many disadvantages, these systems require deep knowledge of the domain, can be time-consuming, difficult to maintain. The machine learning based system learns to make classifications based on past observations. Text classification with machine learning is usually much more accurate than human-crafted rule systems. Deep learning continue to get better with the more data inserted with deep learning algorithms, it requires much more training data than traditional machine learning algorithms. Hybrids systems combines a base classifier trained with machine learning and a rule-based system which can be easily fine-tuned by adding specific rules for those conflicting tags that haven't been correctly modeled by the base classifier.

The choice of machine learning algorithms and classification algorithms depends on the nature of the problem, where extracting features from the text to use in shaping machine and deep learning without losing semantics efficiently is the key to success when working on text classification and similarity check problems, this can be achieved using a word that includes deep learning techniques.

Chapter 3

Theoretical Framework

3.1 Introduction

Optimizing research results has a positive impact on both the searchers and the authors alike, by saving time and effort and by displaying the most relevant results with search query for searchers. In addition, it optimizes the ranking score in search engines and improves the overall quality of available metadata, which enables the author to add as much relevant information to the publication's metadata in order to enhance retrieval, it takes advantage of classifiers and word similarity check to suggest labels to the publications or text, without overburdening the author or indexer

This chapter contains the necessary related theoretical information needed to understand the idea of this thesis, such as related concepts, models, and theories related to the research problem.

Defining and justifying the reasons behind using such an approach for solving the research problem. To achieve thesis goals, and answer the research questions, this work will rely on literature reviews to review the related theories and models developed and worked on by others, to convincingly interpret, explain, and generalize from our findings.

This chapter organized in three sections:

- Section 1: Defines main concepts and terms related to text classification concepts such as text classification, metadata, machine learning, NLP, text vectorization, text tokenization, word embedding, transfer learning, and Term Frequency Inverse.
- Section 2: A brief explanation of the theory behind Classical classifiers and Deep learning classifiers such as Naive Bayes, CNN, and DAN.
- Section 3: This section discusses Al-Quds system model.

3.2 Main Concepts and Terms

1. Metadata

Metadata is defined as "structured information that describes, defines, locates, or otherwise makes it easier to retrieve, use or manage an information resource" (Barker and Campbell, 2010). The concept metadata was used by early libraries as an example of metadata applied in the physical model for classifying and categorizing materials based on the title, author, and subject. They can more easily be located on shelves from among many thousands of others, where the primary purpose of metadata remains the same which is to allow to locate the resources that the researchers looking for efficiently.

2. Machine Learning (ML)

Machine learning is an application of artificial intelligence, where the computer has the ability to learn and improve without being explicitly programmed or have any human intervention (Simeone, 2018).

There are three main types of ML:

1. Supervised: Teach or train the machine by using labeled data, Naïve Bayes as an example.
2. Unsupervised: The machine learns how to do something, without prior knowledge, in this case, the label is absent. K means clustering, which is such an example of it, where it simply defines the relationship between supervised and unsupervised in the presence or absence of labels.
3. Reinforcement: Reinforcement learning lies in between supervised and unsupervised learning. Unlike unsupervised learning, some form of supervision exists, but this does not come in the form of the specification of the desired output for every input in the data. Instead, a reinforcement learning algorithm receives feedback from the environment only after selecting an output for a given input or observation (Simeone, 2018).

3. Text Tokenization

The process of splitting text into pieces called tokens, which often loosely referred to terms or words that is a crucial step in NLP since the meaning of text can easily by analyzing words or sentences in the text. it is used into count numbers of words in the text, and to count the frequency of words (Abbas, et al, 2019).

4. Natural Language Processing

Natural language processing (NLP) is filled of machine learning. It’s a way of analyzing texts by computerized means to perform tasks such as text classification, similarity check, information extracting, which enables computers to understand and process human language (Joseph, et al, 2016), to complete NLP tasks techniques such as syntax which refer to the arrangement of words in text and semantics which refers to meaning of words based on context are used by NLP (Wang, 2009), (Sebastiani, 2002).

5. Vector Representations of Text Data

For humans, understanding text data that is written in natural language can be relatively an easy task, rather than computer which assumed as a hard task, The NLP task allows computers to understand text by helping data processing including converting text into other formats such as vectors. This process is called vectorization (The Stanford Natural Language Processing Group 2015), (Grzegorczyk, 2019). Vectors combine to form vector space, there are different approaches of vectorization, classical approaches to NLP such as one-hot encodings which have a structure and semantic limitations such as capture syntactic meaning relationships across collections of words in vector space, the text input and output are represented as a fixed-length feature vector, one of the most common fixed-length features is bag of words (BOW) as special case of n-gram, where number of word =1.

Bag of Words model is used to present the text for ML algorithms and it’s a way for extracting features from the text to use in ML by converting it into a bag of words, which keeps a count of the total occurrences of most frequently used words. BOW still has its limitations such as losing the ordering of the words, and ignoring semantics of the words (Le and Mikolov, 2014). the following example will illustrate the BOW mechanism to represent a sentence as a bag of words vector: "metadata is a data about data ", the first step is to tokenize words where n=1 ("metadata"," is"," a ","data"," about"," data"), creating a vocabulary of one word called gram, two word called bigram, then create vectors by converting text to be used by the machine learning algorithm (" a ","data"," about","data") for the vocabulary dictionary “the definition of metadata is data about data ” (0,0,0,1,1,2,1,) instead of counting words other method used by calculating the weight of word by statistical evaluate the importance of word corpus term frequency and inverse document frequency (TF-IDF).

6. TF-IDF

Term Frequency Inverse Document Frequency compute a weight to each word, which implies the significance of the word in the document and corpus (Ramos, 2003), (Lilleberg, et al, 2015).

TF(t) = (how many time term X appears in a document) / (Total number of terms in the document).

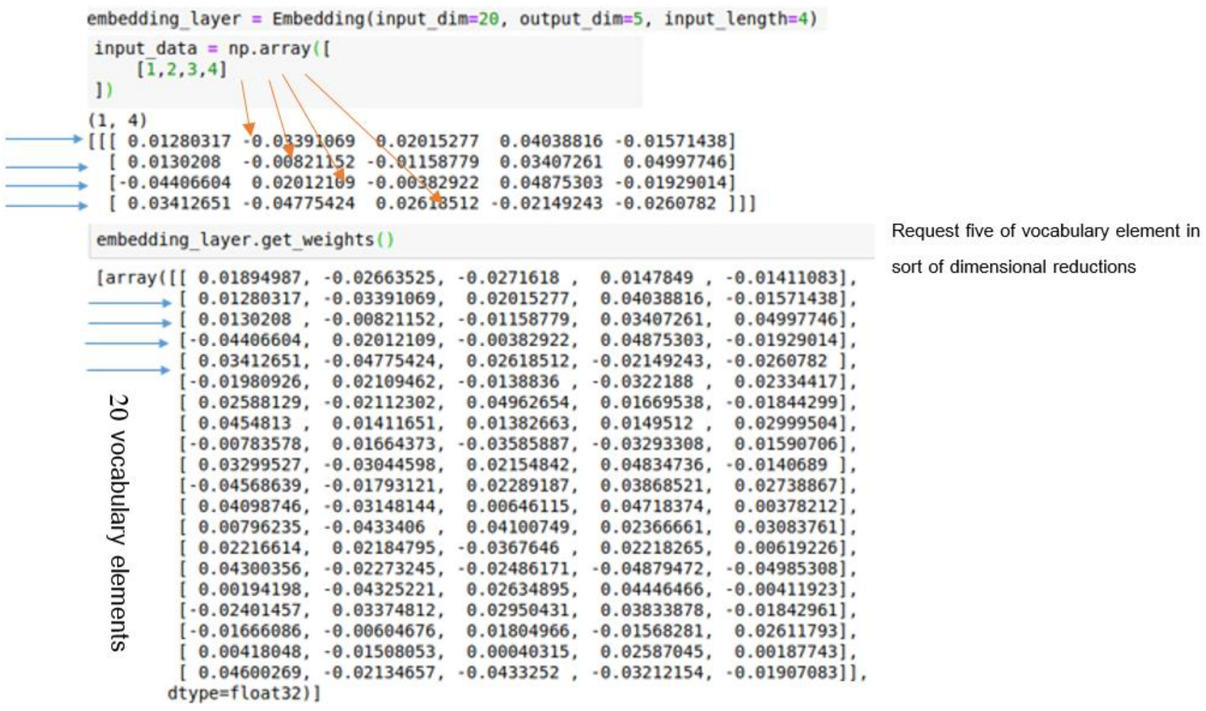
IDF: Inverse Document Frequency, which measures how important a term is.

$$TF - IDF = Tf * IDF$$

(3.1)

7. Word Embedding

Instead of using character encoding by passing each character to the networks that make the network learn a lot about languages, the words encoding will use to achieve the best results. Its type of word representation by converted text into numbers each word mapped to vectors of dimensionality d, the mapping process between integer indices which stand for specific words to dense vectors which stand for their embedding’s handled by embedding layer similar word will therefore be close to each other in the vector space The similarity can be measured in Euclidean distance between vectors, using word embedding enabled machine learning, Deep learning algorithm to process NLP tasks by work with numbers instead of text (Kocmi and Bojar, 2017), in other words Text Vectorization it’s a process of converting a non-numerical text into numerical features which can fed into ML algorithms and applying algebraic operations such as compute the distance between words to compute the similarity. Bag of words, TFIDF, Word2vec can be used to convert text into numerical values. Fig. (3.1) summarize the result of random weights of embedding layer as a table.



```
embedding_layer = Embedding(input_dim=20, output_dim=5, input_length=4)
```

```
input_data = np.array([
    [1,2,3,4]
])
```

```
(1, 4)
[[[ 0.01280317 -0.03391069  0.02015277  0.04038816 -0.01571438]
  [ 0.0130208  -0.00821152 -0.01158779  0.03407261  0.04997746]
  [-0.04406604  0.02012109 -0.00382922  0.04875303 -0.01929014]
  [ 0.03412651 -0.04775424  0.02618512 -0.02149243 -0.0260782 ]]]
```

```
embedding_layer.get_weights()
```

```
[array([[ 0.01894987, -0.02663525, -0.0271618 ,  0.0147849 , -0.01411083],
       [ 0.01280317, -0.03391069,  0.02015277,  0.04038816, -0.01571438],
       [ 0.0130208 , -0.00821152, -0.01158779,  0.03407261,  0.04997746],
       [-0.04406604,  0.02012109, -0.00382922,  0.04875303, -0.01929014],
       [ 0.03412651, -0.04775424,  0.02618512, -0.02149243, -0.0260782 ],
       [-0.01980926,  0.02109462, -0.0138836 , -0.0322188 ,  0.02334417],
       [ 0.02588129, -0.02112302,  0.04962654,  0.01669538, -0.01844299],
       [ 0.0454813 ,  0.01411651,  0.01382663,  0.0149512 ,  0.02999504],
       [-0.00783578,  0.01664373, -0.03585887, -0.03293308,  0.01590706],
       [ 0.03299527, -0.03044598,  0.02154842,  0.04834736, -0.0140689 ],
       [-0.04568639, -0.01793121,  0.02289187,  0.03868521,  0.02738867],
       [ 0.04098746, -0.03148144,  0.00646115,  0.04718374,  0.00378212],
       [ 0.00796235, -0.0433406 ,  0.04100749,  0.02366661,  0.03083761],
       [ 0.02216614,  0.02184795, -0.0367646 ,  0.02218265,  0.00619226],
       [ 0.04300356, -0.02273245, -0.02486171, -0.04879472, -0.04985308],
       [ 0.00194198, -0.04325221,  0.02634895,  0.04446466, -0.00411923],
       [-0.02401457,  0.03374812,  0.02950431,  0.03833878, -0.01842961],
       [-0.01666086, -0.00604676,  0.01804966, -0.01568281,  0.02611793],
       [ 0.00418048, -0.01508053,  0.00040315,  0.02587045,  0.00187743],
       [ 0.04600269, -0.02134657, -0.0433252 , -0.03212154, -0.01907083]],
      dtype=float32)]
```

Request five of vocabulary element in sort of dimensional reductions
20 vocabulary elements

Figure 3.1: Random weights
Embedding layer as lookup table.

8. Glove and Word2vec

Glove is one of the most popular unsupervised learning of word representations word embedding methods proposed in 2013 (Mikolov, et al. 2013), while word embedding converts the text into numeric vectors, the word2vec converts words into vector representations, reconstruct the linguistic context of words based on distance between these vectors, the main purpose of it is to group the output of vectorizing process of similar words together in vector space by reconstructing the linguistic context of words. There are two main learning algorithms, continuous bag-of-words, and continuous skip-gram, Fig.(3.2) represent the CBOW and Skip-gram, where the CBOW predict the current word based on context, the Skip-gram predict the surrounding word of a given word (Jang, et al, 2019), (Lilleberg, et al, 2015), in other words, its predicts word context by using one of its model CBOW or Skip-gram instead of using word-word co-occurrence counts.

The global vectors is used for word representation where it's based on word co-occurrence in corpus the Glove based onto two steps (Naili, et al, 2017), The first step is the construction of a co-occurrence matrix *X* from a training corpus where: $Xi\ j$ is the frequency of the word *i* co-occurring with the word.

$$Xi\ j \ = \sum_{k}^{v} Xi\ k \qquad (3.2)$$

The second step is the factorization of X in order to get vectors that with each word pair of word *i* and word *j*.

$$\vec{w}\,Ti\vec{w}\,j+bi+bj = \log Xij \qquad (3.3)$$

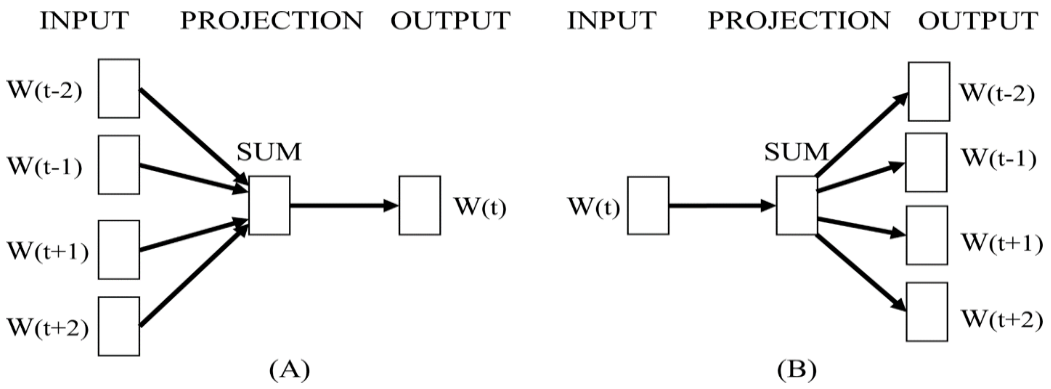


Figure 3.2: Word2vec Model architecture of (A) CBOW and (B) Skip-gram.

Source: (Jang, et al, 2019)

The overall accuracy of using Glove is better than using CBOW or Skip-gram in word2vec (Pennington, 2014 the global vectors for word representation Glove will be used in this work.

9. Transfer Learning

In ML domain, transfer learning is simply learning new tasks that rely on the previously learned tasks (Cer, et al, 2018), or use what acquire as knowledge during training about one task and utilize same way to solve other related tasks. This concept use especially in word embedding where it is not necessary to retrain words and sentence for embedding especially for the task that starves for testing data.

10. Semantic Similarity

Textual semantic similarity measurement calculates the similarity between words, phrase, or texts, which have the same meaning, but which are not lexicographically similar (Martinez-Gil, 2014).

3.3 Classical and Deep Learning Classifiers

Text classification or categorization is the task of assigning set labels to text. In other words, text classification is the process where classifier dividing the set of input document or text into two or more classes in which each document can belong to multiple classes (Sebastiani, 2002), text classification model consists four phases (Ikonomakis, et al, 2005)

1. Text preprocessing: processing include tokenize text, remove punctuation marks, white space, stop word, stemming, and lemmatization.
2. Feature extraction: To represent the text in numerical form by use BOW and TF-IDF in classical model, word2vec and Glove for deep learning model.
3. Machine learning algorithms: there are various approach depending on problem space and available data.
4. Evaluation: There are various methods to determine models effectiveness such as confusion matrix.

Fig (3.3) summarized the text and document classification process, text. Feature extraction

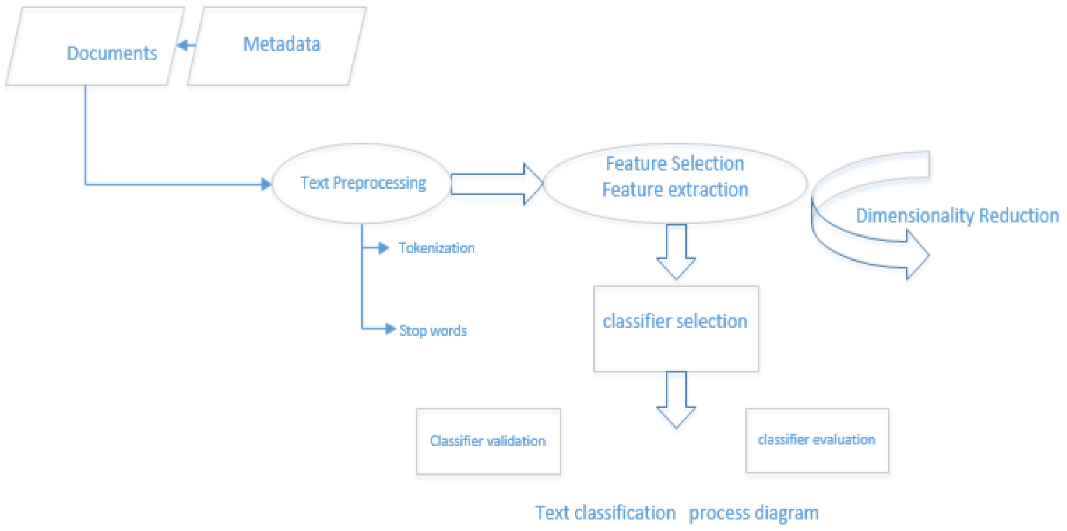


Figure 3.3: Text classification Process

3.3.1. Multinomial Naïve Bayes

The multinomial Naïve Bayes (MNB), is the simplest probabilistic classifier use Bayes theorem to calculate the probability of document *d* belong to class *C*, Pro(C|d), the output tag with the highest one (Ting, et al ,2011), (Abbas, et al, 2019), Fig (3.4) summarize the classical ML model.

$$p(cj|d) = p(d|cj)p(cj)/p(d) \qquad (3.\ 4)$$

The multinomial Naive Bayes classifier (MNB) is one Naive Bayes classifier, it often used as a baseline in text classification. The feature extracting is important to map textual data to real value vectors, text dataset need to convert into numbers to enable Ml digest the input. Multinomial assume each feature have independent distribution, equation (3.5) expressed the probabilities. Calculating the prior probability of each tag *d* in class *cj*, using Laplace smoothing (Feng and Xiaoqing, 2007) if probability comes out to be zero, finally multiply all the probabilities to get the bigger.

$$p(d|cj) = p(d1|cj) * p(d2|cj) * \dots \dots * p(dn|cj) \qquad (3.5)$$

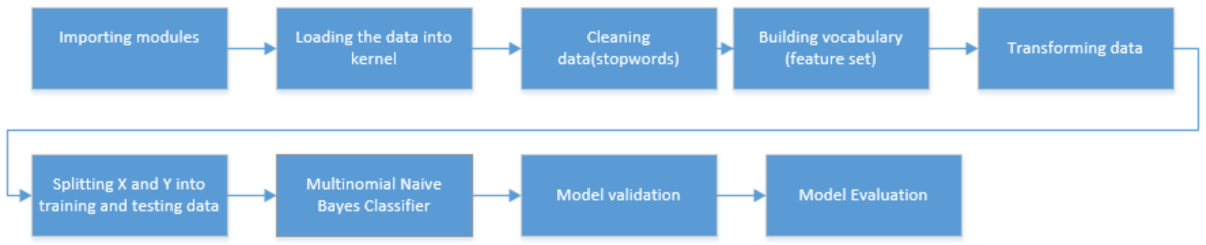

Importing modules
Loading the data into kernel
Cleaning data(stopwords)
Building vocabulary (feature set)
Transforming data
Splitting X and Y into training and testing data
Multinomial Naive Bayes Classifier
Model validation
Model Evaluation

Figure 3.4: Classical ML Model

3.3.2. Convolutional Neural Network

Convolutional neural network (CNN) is a type of deep learning neural network, typically use for image processing. Applying CNN to text with matrix of word vectors instead of image pixel (Amin and Nadeem, 2018), the text represented by one-dimensional array with 1dimention convolutional, and pooling operations. Fig (3.5) show the result of applying 1 dimension convolution.

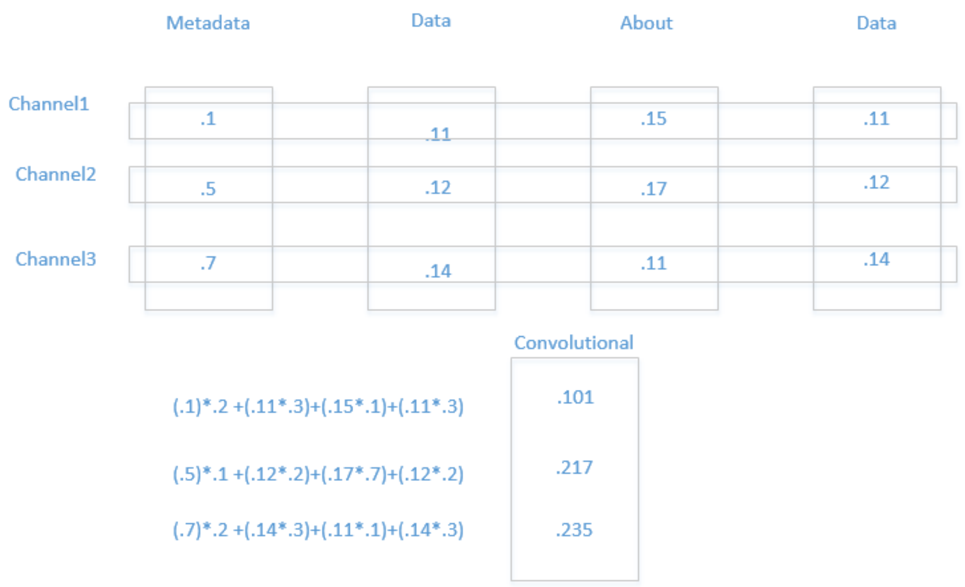


Figure 3.5: Applying 1D convolution.

CNN uses operations called convolution and pooling, the convolution is an act of taking the original data and creating an original map from it, convolution is a special type of linear operation between two functions that expresses how the shape of one function is affected by the other. The basic block of CNN contains a convolution layer the hidden layer (the filter) which acts as feature extractor. It can detect complex pattern when it works with sequential data like text by using a one-dimensional convolutions to generate a matrix, this matrix run through activation layer which introduces a nonlinearity by using activation function such as (ReLu) function f(x) = max(0,x).

Pooling layer reducing the matrix size or reduces the dimensionality of feature map by combining the vectors resulting from different convolutional windows into single dimension vector. This is done by taking of different types such as max, average or sum, the two processes described above convolutional and pooling the features extractor pass these features as a vector of one row to the network to be trained for classification the fully connected layer its output list of possible class label attach to document where it performs the prediction (Lilleberg, et al, 2015), (Martinez-Gil ,2014).

$$o_j = f\left(\sum_i w_{i,j} a_i + b_i\right) \quad (3.6)$$

Neural network formula

Where the input is layer *a* , the output is layer *o*. To calculate the values for each output node, multiplying each input node by a weight *w,* the algorithm starts by initializing the weights with random values and they are then trained with backpropagation by using optimization methods to reduce the error between the computed and the desired output, and add a bias *b.* the activation function *f* is applied for the summation of all these variables. Fig (3.6) show the general model of the CNN that consists of three main layers:

1. Input layer.
2. Hidden layers
3. Output layers.

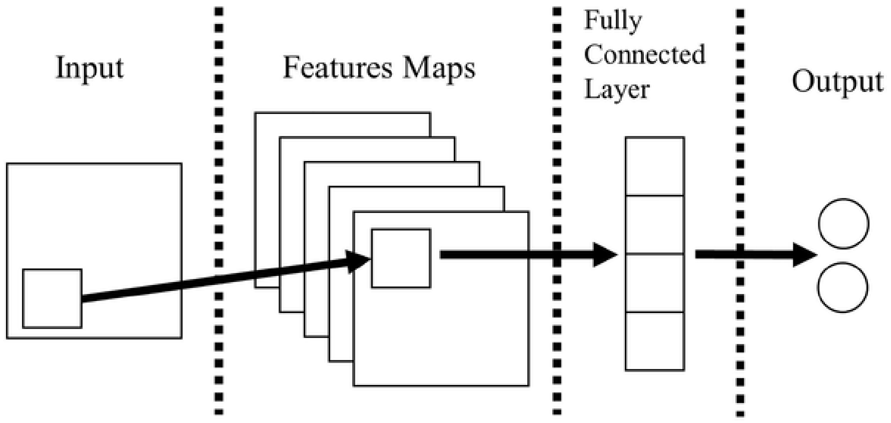


Figure 3.6: The general model of the CNN

Source(Jang, et al 2019)

Within the input layer, parsed the data is passed to the features maps without losing semantic of words in a sentence, followed by conventional and max pooling, until CNN creates fully connected layer that combines all convolutional and pooling layers, which output the results to output layer(Jang, et al, 2019).

Fig. (3.7) show the model of CNN with characters based encoding, that approach force the network to learn too much about language.

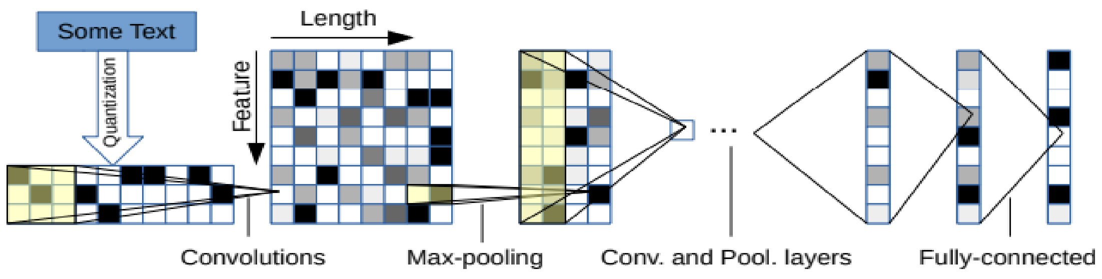


Figure 3.7: CNN Model for Sentence Classification based on character encoding

Source: (Zhang, et al, 2015)

Fig. (3.8) represent the model of CNN based on word encoding, staring with the first layer which embed the word into low-dimensional vectors by using word2vec or Glove, the next layer map the features without losing the semantics of words, followed by conventional and max pooling until CNN creates fully connected layer that combines all convolutional and pooling layers, which output the results to output layer. The document classification task done by learned jointly between word embedding and neural model, the embedding layer used in front end of neural network and fit in a supervised way using the backpropagation methods, for learning word embedding from corpus methods such word2vec and Glove.

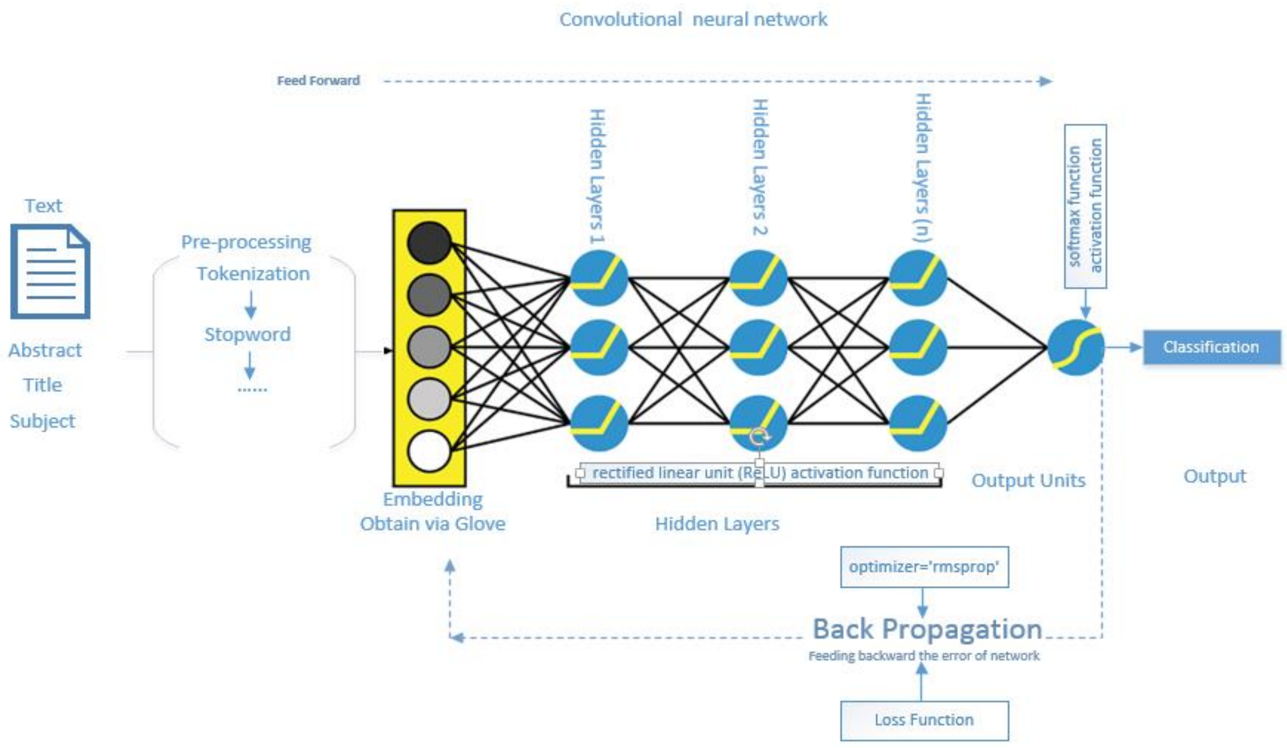


Figure 3.8: CNN classification model.

3.3.3. Deep Averaging Network

Deep averaging network (DAN) is a simpler model of neural network, which use the average of word embedding of input text, feeds the average through several layers with non-linearity. DAN ignores the ordering of the words in which instead of dropping individual units, it drops the entire word tokens from the input (Cer, et al, 2018). "The intuition behind deep feed-forward neural networks is that each layer learns a more abstract representation of the input than the previous one"(Iyyer, et al, 2015). Using DAN based on universal encoding models provide sentence level embedding and give superior results for sentences similarity task. It does not need a huge amount of data, it only needs to download the universal encoder and make some modifications for it. Fig.(3.9) shows the key idea of DAN, it takes the input *Z* whether it is a text or a document to produce the output, where in between them it represents a word as continues bag of word in 300 dimensions of embedding vectors by adding them together.

DAN, representing words as continues bag of words, each embedding is a dimensional vector. Taking this representation and adding them together and applying linear transformation then pass it to non-linearity element, these nonlinearity put on top of simple of average words for statement analysis. Finally apply the softmax classifier on the output.

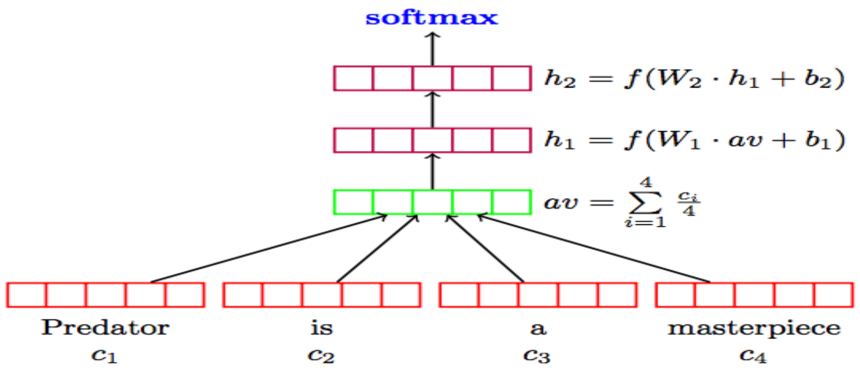

softmax
$h_2 = f(W_2 \cdot h_1 + b_2)$
$h_1 = f(W_1 \cdot av + b_1)$
$av = \sum_{i=1}^{4} \frac{c_i}{4}$
Predator c_1
is c_2
a c_3
masterpiece c_4

Figure 3.9: Deep averaging network
Source (Iyyer, et al, 2015)

$$z0 = CBOW(w1 \ldots\ldots, wn) \sum_{i}^{n} E[wi] \tag{3. 7}$$

$$z1 = g(w1z0 + b1) \tag{3. 8}$$

Then take the previous representation of document and applying linear transformation of it defining by the variable $w\ and\ b$ and do this again for it.

$$z2 = g(w2z1 + b2) \tag{3.9}$$

Finally take the output y:

$$y = softmax(z2) \tag{3.10}$$

3.4 Al-Quds System

Al-Quds System is a framework use the automatic classification approach to assist authors and indexer to classify publications, edit publication metadata and compute the relevancy between document subject and description based on word similarity model. Searchers may use different words or synonyms to search for a specific topic, The displayed result depends on the methods and algorithms used by the search engine, the display results maybe not relevant to search query, the classification process of publications are carried out by the authors or indexers, who in turn may use different words to describe publications in the metadata. To overcome this problem, Al-Quds System extends features of Dublin Core to allow the author or indexers to use synonyms words to describe their publications, based on the content of abstract, title, and subject. After suggesting several words to use and calculating the degree of similarity between the proposed words for title, subject and the abstract a mathematical process apply to combine words into single vector (Iyyer, et al, 2015), the next step is to suggest categories based on these keywords.

Al-Quds System uses two methods to find the similar words of the root word. The first method is done by using word2vec algorithms such as CBOW and Skip-gram to find the relationships between words.

CBOW forms sets of word used to predict similar word, The Skip-gram algorithm work on the opposite way of CBOW, whereby using the root word to find the surrounding word as shown in fig. (3.10). This approach is used to find the similar or surrounding words of the root word, by converting words into vectors to find the distance between those vectors, the closest distance means more relevant words. This method allows performing some algebraic operation on words such as subtracting two words. The difference between two vectors carry some meanings (Jang, et al, 2019).

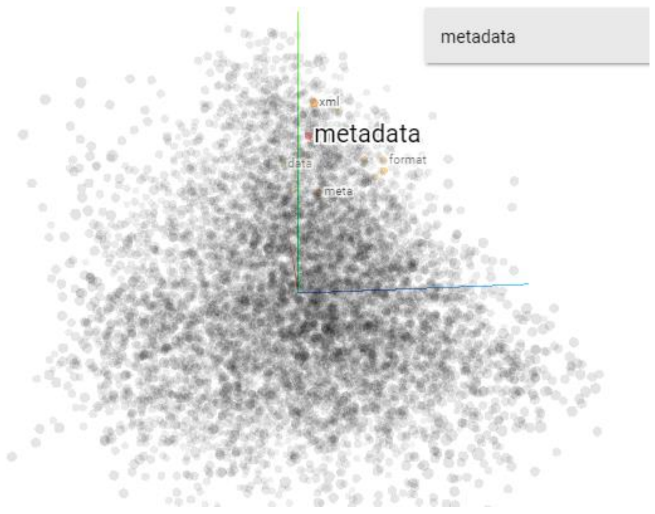


Figure 3.10: Example of word similarity
Source**:** (Embedding Projector - Visualization of High-Dimensional Data, n.d.)

The second method, by computing the cosine similarity between vectors. Very similar words negatively affect the precision where small angle produce almost the same similarity. To overcome this problem equation (3.11) used to calculate the similarity by using angular similarity to distinguishes nearly parallel vectors (Cer, et al, 2018).

$$\text{sim}(\mathbf{u},\mathbf{v}) = \left(1 - \arccos\left(\frac{\mathbf{u}\cdot\mathbf{v}}{||\mathbf{u}||\ ||\mathbf{v}||}\right)/\pi\right) \quad (3.11)$$

Source (Cer, et al, 2018)

Instead of training text for embedding which consuming resources and time also need huge amount of text, the google universal sentence encoder (Cer, et al, 2018) used in the similarity check model (Universal-Sentence-Encoder | TensorFlow Hub, n.d.).

Fig. (3.11) show the result of using Glove framework to learn the word vector for ("data", "about", "data") to predict the fourth word ("Metadata").

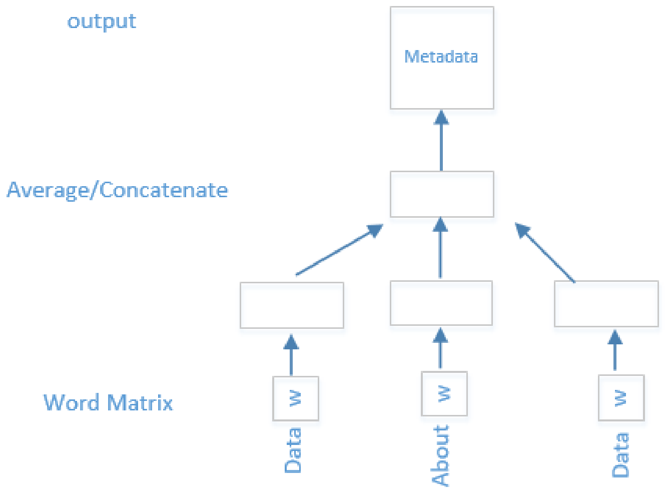


Figure 3.11: Using Glove to predict next word

3.4.1. Al-Quds System Model

Al-Quds System consists of several model, Fig (3.12) show a process flow diagram of Al-Quds System model, this model works together to increase the relevance between display search results and search query through main functions. Fig (3.13) show the prototype of Al-Quds System.

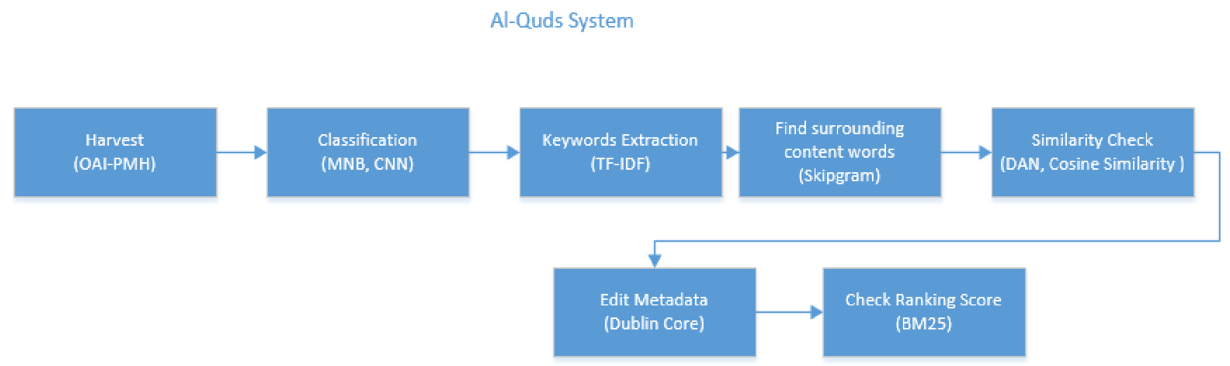
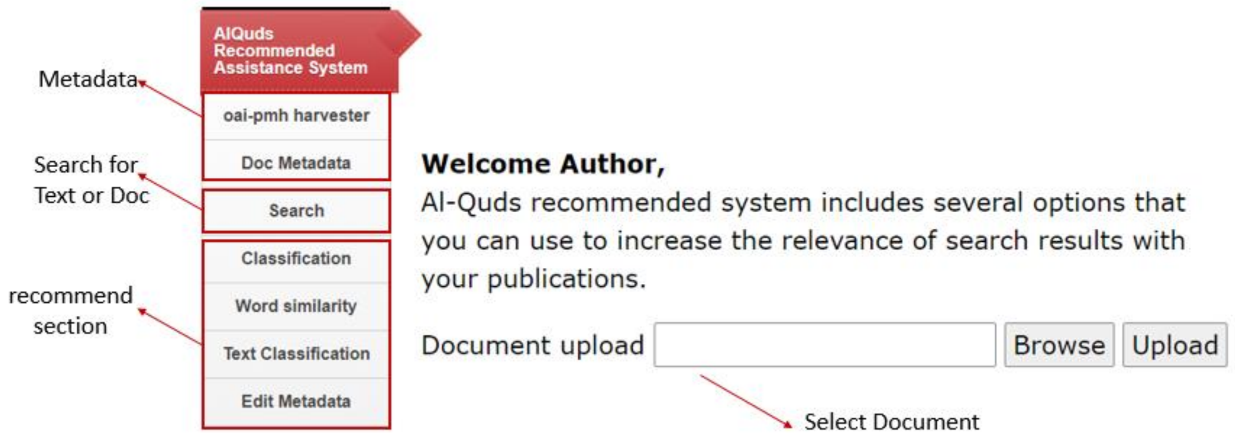

Al-Quds System
Harvest (OAI-PMH)
Classification (MNB, CNN)
Keywords Extraction (TF-IDF)
Find surrounding content words (Skipgram)
Similarity Check (DAN, Cosine Similarity)
Edit Metadata (Dublin Core)
Check Ranking Score (BM25)

Figure 3.12: Al-Quds System Process

.



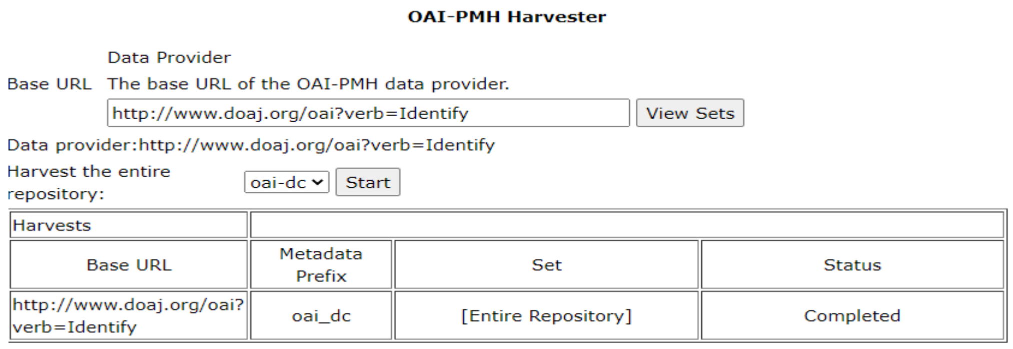
Welcome Author,

Al-Quds recommended system includes several options that you can use to increase the relevance of search results with your publications.

Figure 3.13: A prototype of Al-Quds System

3.4.1.1. The Harvester Model

This model is responsible for harvesting metadata and publications from different open access repositories using OAI-PMH protocol (Awasthi and Jaiswal, 2008), (Kushwaha, 2015). Using OAI-PMH harvester to harvest metadata from different open access digital repository (Roy, et al, 2017), Fig.(3.14) show prototype of OAI-PMH harvester.


OAI-PMH Harvester

Data Provider

Base URL The base URL of the OAI-PMH data provider.

http://www.doaj.org/oai?verb=Identify

Data provider:http://www.doaj.org/oai?verb=Identify

Harvest the entire repository: oai-dc

Harvests			
Base URL	Metadata Prefix	Set	Status
http://www.doaj.org/oai?verb=Identify	oai_dc	[Entire Repository]	Completed

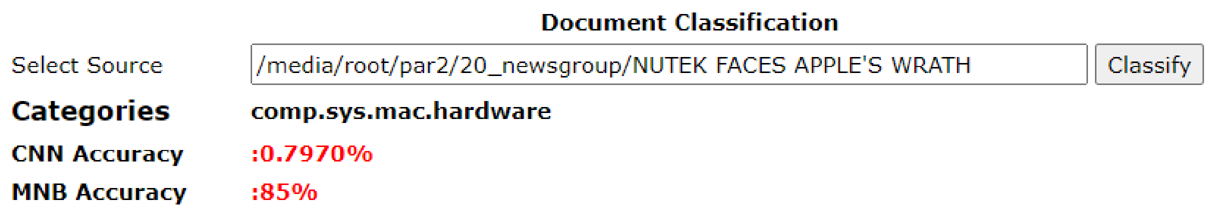
Figure 3.14: A prototype of OAI-PMH harvester

3.4.1.2. Classification Model

This model is responsible for text and document classification. The CNN document classification model is to use an embedding layer as input, followed by a one-dimensional convolutional neural network, pooling layer, and then a prediction output layer. Using deep learning classifiers such as CNN on top word embedding will exploit the capabilities of CNN for text classification (Amin and Nadeem, 2018).

Classical machine learning such as Multinomial Naïve Bayes used to test the classification result and performance, for the similarity check, which will use for assistance system google sentences encoder was used to encoding sentences into embedding vectors that specifically target transfer learning to other NLP tasks based on DAN and use BM25 function to compute the relevance.

Document classification is responsible for documents classification by using CNN classifiers or multinomial naive Bayes classifier, fig. (3.15) show a prototype of the classifiers.


Document Classification

Select Source	/media/root/par2/20_newsgroup/NUTEK FACES APPLE'S WRATH
Categories	comp.sys.mac.hardware
CNN Accuracy	:0.7970%
MNB Accuracy	:85%

Figure 3.15: A prototype of classifiers model

3.4.1.3. Word Similarity Check Model

This model is responsible for word similarity check to find the nearest words by using principal component analysis (PCA) or semantic similarity, both on words level and sentences level by computing the representation of each message in same embedding size.

This model use a pre-trained universal sentence encoder with deep average network with universal sentence encoder used to compute the similarity between the abstract, subject and title. Fig. (3.16) show the height level of the semantic similarity.

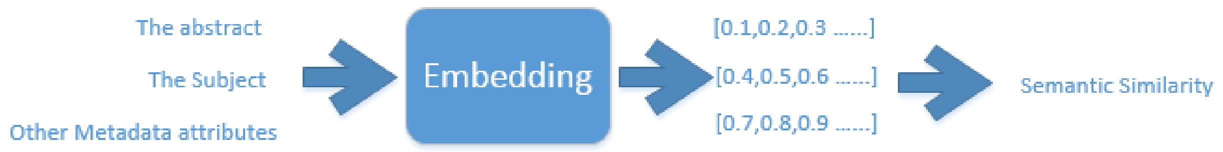


Figure 3.16: High level of the semantic similarity
Source:(Cer, et al. 2018)

Fig. (3.17) show the result of using the universal sentence encoder for semantic similarity checks on sample word such as metadata and its relevant keywords.

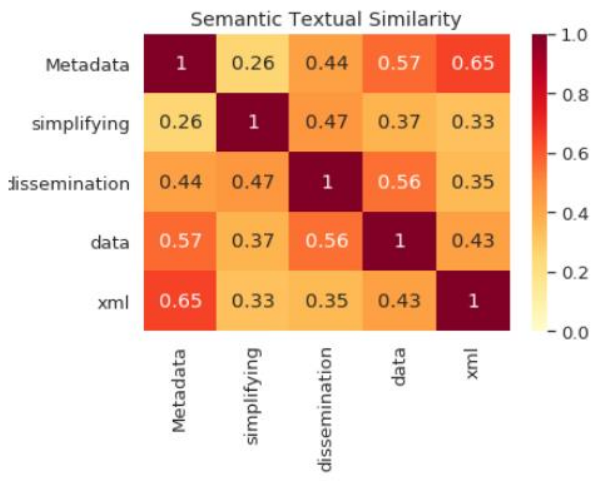
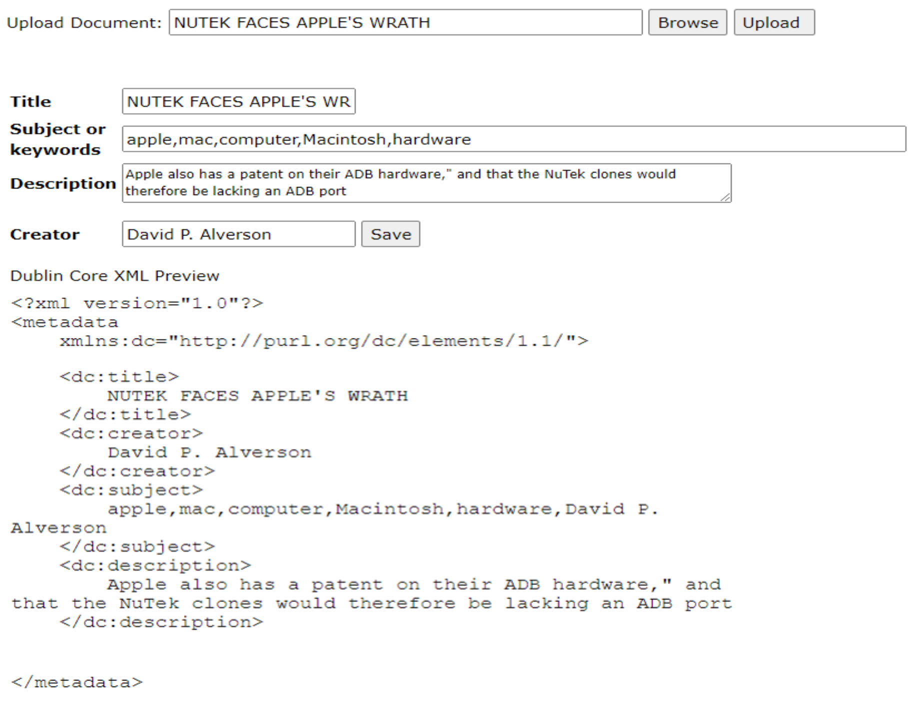


Figure 3.17: Sematic textual similarity heat map

3.4.1.4. Metadata Editing Model

This model responsible for editing metadata allows authors and indexers to edit document metadata based on the keywords and categories where they got it from word similarity check and text classification, Fig. (3.18) shows a prototype of metadata editing.



Upload Document:	NUTEK FACES APPLE'S WRATH
Title	NUTEK FACES APPLE'S WR
Subject or keywords	apple,mac,computer,Macintosh,hardware
Description	Apple also has a patent on their ADB hardware," and that the NuTek clones would therefore be lacking an ADB port
Creator	David P. Alverson

```
<?xml version="1.0"?>
<metadata
    xmlns:dc="http://purl.org/dc/elements/1.1/">

    <dc:title>
        NUTEK FACES APPLE'S WRATH
    </dc:title>
    <dc:creator>
        David P. Alverson
    </dc:creator>
    <dc:subject>
        apple,mac,computer,Macintosh,hardware,David P.
Alverson
    </dc:subject>
    <dc:description>
        Apple also has a patent on their ADB hardware," and
that the NuTek clones would therefore be lacking an ADB port
    </dc:description>


</metadata>
```

Figure 3.18: A prototype of metadata editing

3.4.1.5. Ranking Model

The ranking model computes the relevance score of search terms and the document or corpus. This

uses best matching function (BM25) (Robertson and Zaragoza, 2009). To compute the scores search query *Q* with each document *D*, its uses TFIDF (Ramos, 2003), to compute the importance of words in document or corpus, where TF represent the term frequency in document, its compute how often the term rises in document and IDF represent the inverse document frequency which reduces the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely.

$$idf(t,D) = log\left(\frac{N}{count(d \in D : t \in d)}\right) \quad (3.12)$$

$$tf(t,d) = log(1 + freq(t,d)) \quad (3.13)$$

$$\text{BM25}(D,Q) = \sum_{i=1}^{n} IDF(q_i, D) \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i) + k_1 \cdot (1 - b + b \cdot |D|/d_{avg}))} \quad (3.14)$$

- IDF : is the inverse of term q in document D
- F(qi ,D):is the Repetition of term qi in document D
- |D|: number of words in Document D
- d avg : average number of words in document D
- b and K1 are hyper parameters of BM25 their value will be (.05 < b<1.2 and 1.2<k1<2)(Robertson and Zaragoza 2009).

Chapter4

Experimental Results and Discussion

4.1 Introduction

This chapter aims at discussing the implementation of the proposed framework, which fully explained in Chapter 2 and 3. Moreover, it lists the evaluation metrics that used to evaluate the efficiency of the proposed framework. It also presents and analyses the experimental results of the proposed text classification mechanism, word similarity check mechanism, and finally, a comparative analysis between the results of the proposed framework models.

4.2 Dataset Preprocessing

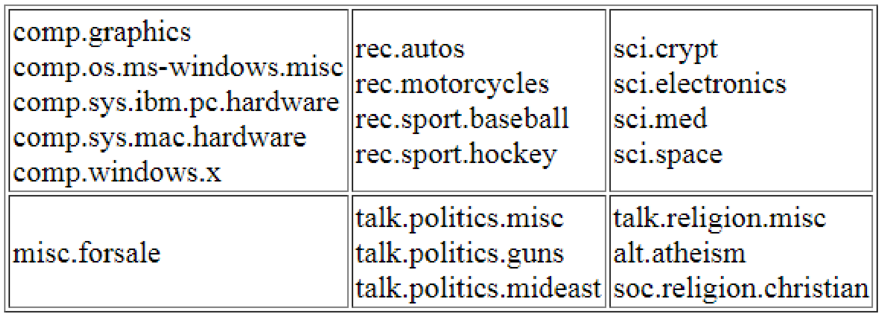
4.2.1.   Data Collection

To verify the validity of the proposed framework, the experiment used commonly used publicly available datasets the 20 newsgroups dataset ("Home Page for 20 Newsgroups Data Set" n.d.). Collected by Ken Lang.

4.2.2.   Labels Distribution

The 20-newsgroups dataset is a popular dataset for experiments in-text application of ML techniques, such as text classification. The public 20- newsgroups dataset collection contains approximately 20,000 records normalized and consistent, portioned almost equally across 20 different categories. Some of its categories very related to each other, while others are unrelated. Table. (4.1) shown a list of the 20 newsgroups categories, partitioned according to the subject matter.

Table 4.1: List of the 20 newsgroups



comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

Source of text (Home Page for 20 Newsgroups Data Set, n.d.)

4.2.3. Feature Extraction

Since the dataset is a series of words, its need to clean and tokenize then split into the train, the test set with a ratio of 75% for training, and 25% for testing. For features extraction, then convert into numerical representations before running ML algorithms by using TF-IDF or BOW for the classical model, and convolution layers followed by max-pooling and an activation function used for the deep learning model.,

4.3 Experimental Environment

The experimental machine had CPU i7, 8GB memory and a 256 GB solid-state drive (SSD). The experiment system used Ubuntu operating system with python 2.7.

4.4 Evaluation Criteria

An evaluation criterion is based on:

4.4.1. Confusion Matrix

for the classification model, a number of generic evaluation measures are being used to evaluate machine learning algorithms, specifically, consider training time, accuracy, precision, recall, F-score, computed using the confusion matrix as shown in Figure (4.1).

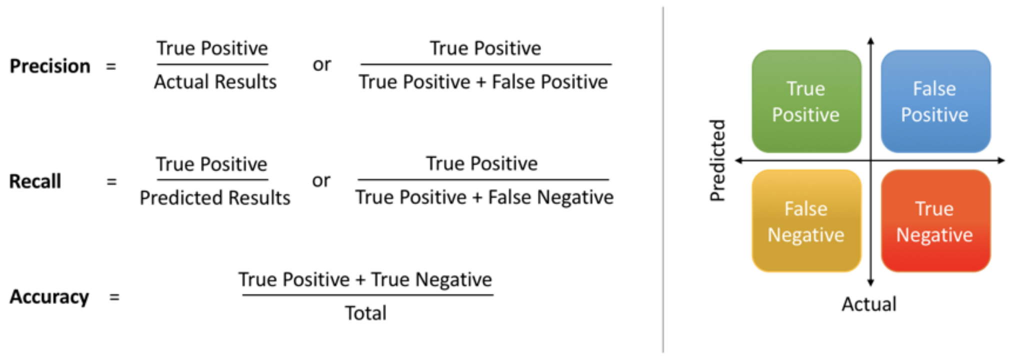

Precision = True Positive / Actual Results or True Positive / True Positive + False Positive
Recall = True Positive / Predicted Results or True Positive / True Positive + False Negative
Accuracy = True Positive + True Negative / Total
True Positive
False Positive
False Negative
True Negative
Predicted
Actual

Figure 4.1: The confusion matrix
Source:(Shrivastav, n.d.)

Referring to Figure (4.1), we define the following concepts.

- True Positive (TP): When outcome is actually positive and predicted positive.
- False Positive (FP): When the outcome is actually negative but predicted to positive.
- True Negative (TN): When the outcome is actually negative and predicted to negative.
- False Positive (FN): When the outcome is actually positive but predicted to negative.
- Accuracy: Accuracy measure percentage value for correct prediction of data. It is calculated by the follow formula:

$$\text{Accuracy} = TP+TN/TP+FP+FN+TN \quad (4.1)$$

- Precision: Precision is the ratio of correctly predicted positive observations of the total predicted positive observations. It is calculated by the follow formula:

$$\text{Precision} = TP/TP+FP \quad (4.2)$$

- Recall: Recall is the ratio of correctly predicted positive observations to all observations in the actual class. It is calculated by the follow formula:

$$\text{Recall} = TP/TP+FN$$

(4.4)

- F1 score: F1 score is the weighted average of Precision and Recall. It is calculated by the follow formula:

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

(4.5)

4.4.2. Cosine Similarity

A cosine similarity use to compute the relevancy of words and sentences, where the closer the cosine value to one, the smaller the angle, the higher the match between words vectors. It is calculated by the follow formula:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

(4.6)

4.4.3. BM25

The BM25 functions use to compute the relevancy between search term and display results. It is calculated by the follow formula:

$$\text{BM25}(D, Q) = \sum_{i=1}^{n} IDF(q_i, D) \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i) + k_1 \cdot (1 - b + b \cdot |D|/d_{avg}))}$$

(4.7)

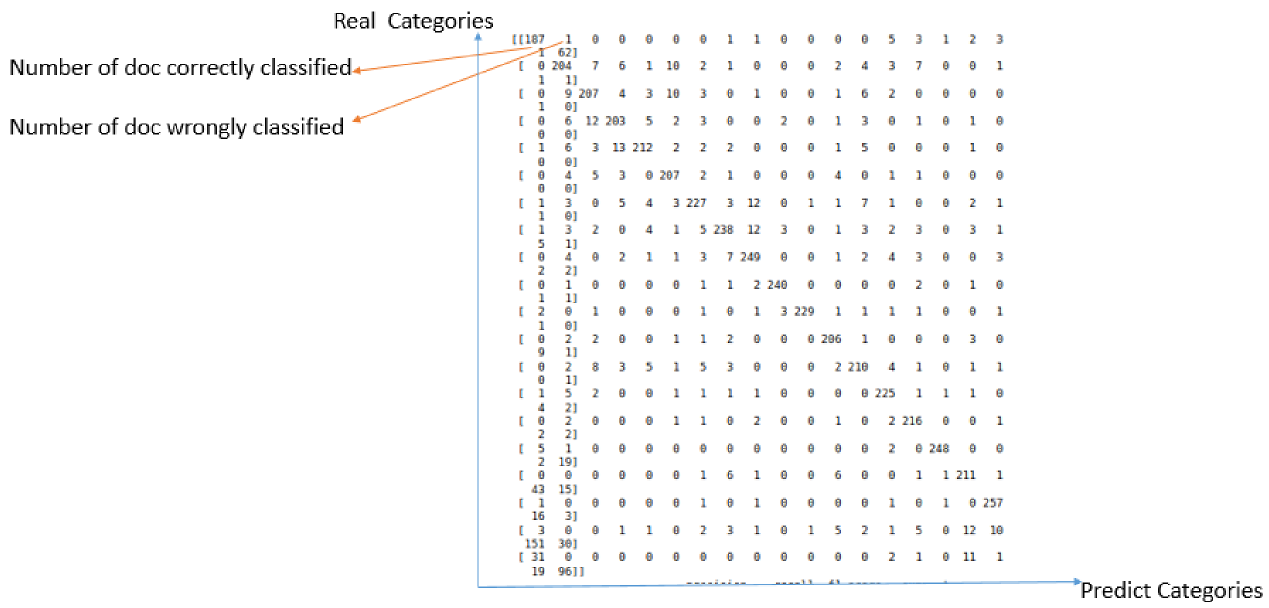
4.5 Experimental Results and Analysis

4.5.1. MNB Experiments

Python programming language was used to implement this model. Multinomial Naïve Bayes normally requires integer feature counts. Classifier starts with creating a list of folder names to make valid, loading the data into kernel, BOW and TF-IDF use for features extraction to build the vocabulary. BOW collect different words that occur the dataset, associated each word with a count of how it occurs in vocabulary, in this method the order does not matter, Fig. (4.2) and Fig. (4.3) show the results of Multinomial Naïve Bayes features extraction with BOW. TF-IDF is another statistical method that can use for feature extraction in Multinomial Naïve Bayes classifier, especially when stop words are not removed (Ramos 2003), which calculates the frequency of a word in a document and reduces the weight of common words. Fig. (4.4) show number of words with frequency, and Fig. (4.5) show the transform a count matrix to a normalized TF-IDF representation.

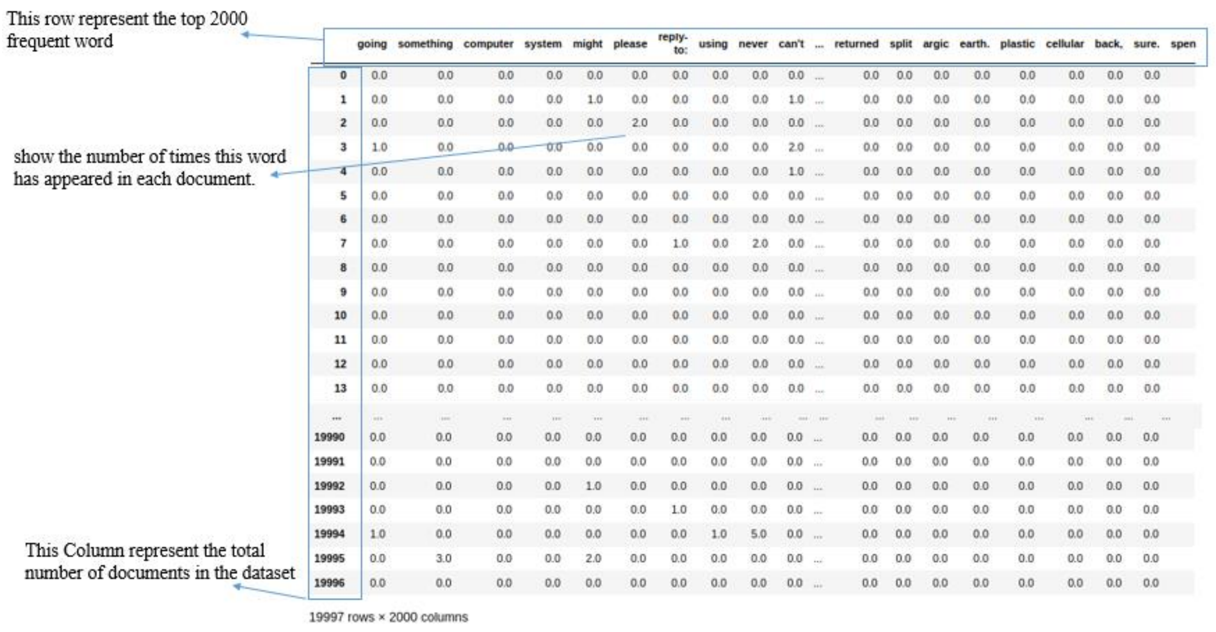
The vectorization process use vocabulary to construct the d-dimensional feature vectors for each document, where the dimensionality is equal to the number of different words in the vocabulary. The feature vectors represent absolute counts since the probabilistic model used for the Naive Bayes classifier is the multinomial. The text preprocessing used to remove stop word and punctuations to make a dimensionality reduction in order to remove noise feature, the model get 390170 vocabulary, building final feature list from vocab by use the top 2000 frequent vocab words as features. Transforming data into X and Y where each row represent one doc and each column represent one word from feature list.

To get nonzero probability, set the alpha or smoothing parameter value =1, this will add 1 to every probability, keep the prior probabilities of each class parameter class_prior to default, to tells the model to learn class prior probabilities, set the fit_proir parameters to default value =True, Table (4.3) list value of MNB parameters. Finally print the confusion matrix and classification report.


Real Categories
Number of doc correctly classified
Number of doc wrongly classified
Predict Categories

```
[[187   1   0   0   0   0   0   1   1   0   0   0   0   5   3   1   2   3
    1  62]
 [  0 204   7   6   1  10   2   1   0   0   0   2   4   3   7   0   0   1
    1   1]
 [  0   9 207   4   3  10   3   0   1   0   0   1   6   2   0   0   0   0
    1   0]
 [  0   6  12 203   5   2   3   0   0   2   0   1   3   0   1   0   1   0
    0   0]
 [  1   6   3  13 212   2   2   2   0   0   0   1   5   0   0   0   1   0
    0   0]
 [  0   4   5   3   0 207   2   1   0   0   0   4   0   1   1   0   0   0
    0   0]
 [  1   3   0   5   4   3 227   3  12   0   1   1   7   1   0   0   2   1
    1   0]
 [  1   3   2   0   4   1   5 238  12   3   0   1   3   2   3   0   3   1
    5   1]
 [  0   4   0   2   1   1   3   7 249   0   0   1   2   4   3   0   0   3
    2   2]
 [  0   1   0   0   0   0   1   1   2 240   0   0   0   0   2   0   1   0
    1   1]
 [  2   0   1   0   0   0   1   0   1   3 229   1   1   1   1   0   0   1
    1   0]
 [  0   2   2   0   0   1   1   2   0   0   0 206   1   0   0   0   3   0
    9   1]
 [  0   2   8   3   5   1   5   3   0   0   0   2 210   4   1   0   1   1
    0   1]
 [  1   5   2   0   0   1   1   1   1   0   0   0   0 225   1   1   1   0
    4   2]
 [  0   2   0   0   0   1   1   0   2   0   0   1   0   2 216   0   0   1
    2   2]
 [  5   1   0   0   0   0   0   0   0   0   0   0   0   2   0 248   0   0
    2  19]
 [  0   0   0   0   0   0   1   6   1   0   0   6   0   0   1   1 211   1
   43  15]
 [  1   0   0   0   0   0   1   0   1   0   0   0   0   1   0   1   0 257
   16   3]
 [  3   0   0   1   1   0   2   3   1   0   1   5   2   1   5   0  12  10
  151  30]
 [ 31   0   0   0   0   0   0   0   0   0   0   0   0   2   1   0  11   1
   19  96]]
```

Figure 4.2: MNB with BOW


This row represent the top 2000 frequent word
show the number of times this word has appeared in each document.
This Column represent the total number of documents in the dataset

	going	something	computer	system	might	please	reply-to:	using	never	can't	...	returned	split	argic	earth.	plastic	cellular	back.	sure.	spen
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
7	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	2.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
19990	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
19991	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
19992	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
19993	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
19994	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	5.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
19995	0.0	3.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
19996	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

19997 rows × 2000 columns

Figure 4.3: MNB features extraction with BOW

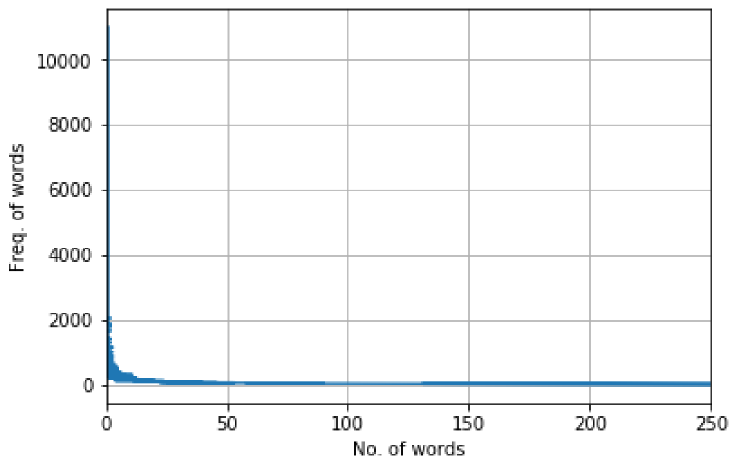
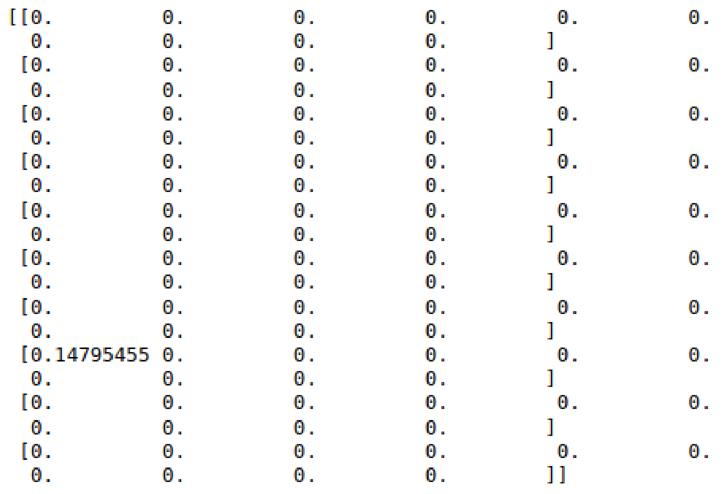


Figure 4.4: Number of words with frequency



```
[[0.         0.         0.         0.          0.          0.
  0.         0.         0.         0.         ]
 [0.         0.         0.         0.          0.          0.
  0.         0.         0.         0.         ]
 [0.         0.         0.         0.          0.          0.
  0.         0.         0.         0.         ]
 [0.         0.         0.         0.          0.          0.
  0.         0.         0.         0.         ]
 [0.         0.         0.         0.          0.          0.
  0.         0.         0.         0.         ]
 [0.         0.         0.         0.          0.          0.
  0.         0.         0.         0.         ]
 [0.         0.         0.         0.          0.          0.
  0.         0.         0.         0.         ]
 [0.14795455 0.         0.         0.          0.          0.
  0.         0.         0.         0.         ]
 [0.         0.         0.         0.          0.          0.
  0.         0.         0.         0.         ]
 [0.         0.         0.         0.          0.          0.
  0.         0.         0.         0.         ]]
```

Figure 4.5: Transform a count matrix to a normalized TF-IDF representation.

4.5.2. CNN Experiments

Python programming language was used to implement this model. This model applied convolutional neural network for NLP for text classification, it is combined with word vectorization and word embedding to find best classes for each documents in the corpus.

A CNN trained on top of pre-trained word vectors obtained from unsupervised neural langue model trained by Glove (Pennington, et al 2014) to obtaining vector representation for words. Its consist of 6B tokens, 400K vocab50d, 100d, 200d, and 300d vectors to obtain better vector representation and improve the accuracy.

Working with sequential data (text), the classification need to use one-dimensional convolution.

CNN uses a python to run an open source neural network library Keras (François Chollet, n.d.). Running on top of end to end open source platform for machine learning Tensor Flow(TensorFlow, n.d.). CNN consist of conventional layer, max pooling layer and dense layer to calculate the output or specify the categories. In other words, output = activation (dot (input, kernel) + bias)). The input represent the input data, kernel represent the weight data ,dot represent dot product of all input and its corresponding weights, bias represent a biased value and activation represent the activation function. The corpus used in this work is 46.7MB from 20-newsgroup, which simulates goals to classify text into different categories, based on word vectors dens. After preprocessing phase 19997 texts founded, 174105 unique tokens got after applying stop words and tokenization found. Got 400000 word vectors by loading embedding weights to the embedding layer. The accuracy evaluated by splitting data into training set and validation set using the same ratio used in the classical model .75% , 14998 sample for training, and 25%, 4999 sample for validation.

The model preparing the embedding matrix by set 0 for all words doesn't found in the embedding index, then load pre-trained word embedding into embedding layer by setting the (trainable number of weights that are not updated during training with backpropagation) parameter to false, to take advantage of transfer learning. In addition, the result compares when the trainable parameter is true.

Fig. (4.6) show the Architecture of proposed CNN, Fig. (4.7) show model summary and the number of parameters available for training with transfer learning trained parameters is true, fig.

(4.8) show model summary and the number of parameters available for training without transfer learning trained parameters is false.

The learning process configured in order to specify the optimizer and loss function according to the following settings to not over fit start train a 1 dimensional CNN 128 filters, the kernel size is 5, and the rectified linear unit (Relu) activation function for hidden layer, Backpropagation methods reduce the error between the computed and the desired output, take advantage of the iterative training process in neural network by using the error computed by Cross Entory loss function who's loss minimize using (RMSprop) optimizer, and finally softmax activation function for the output layer.

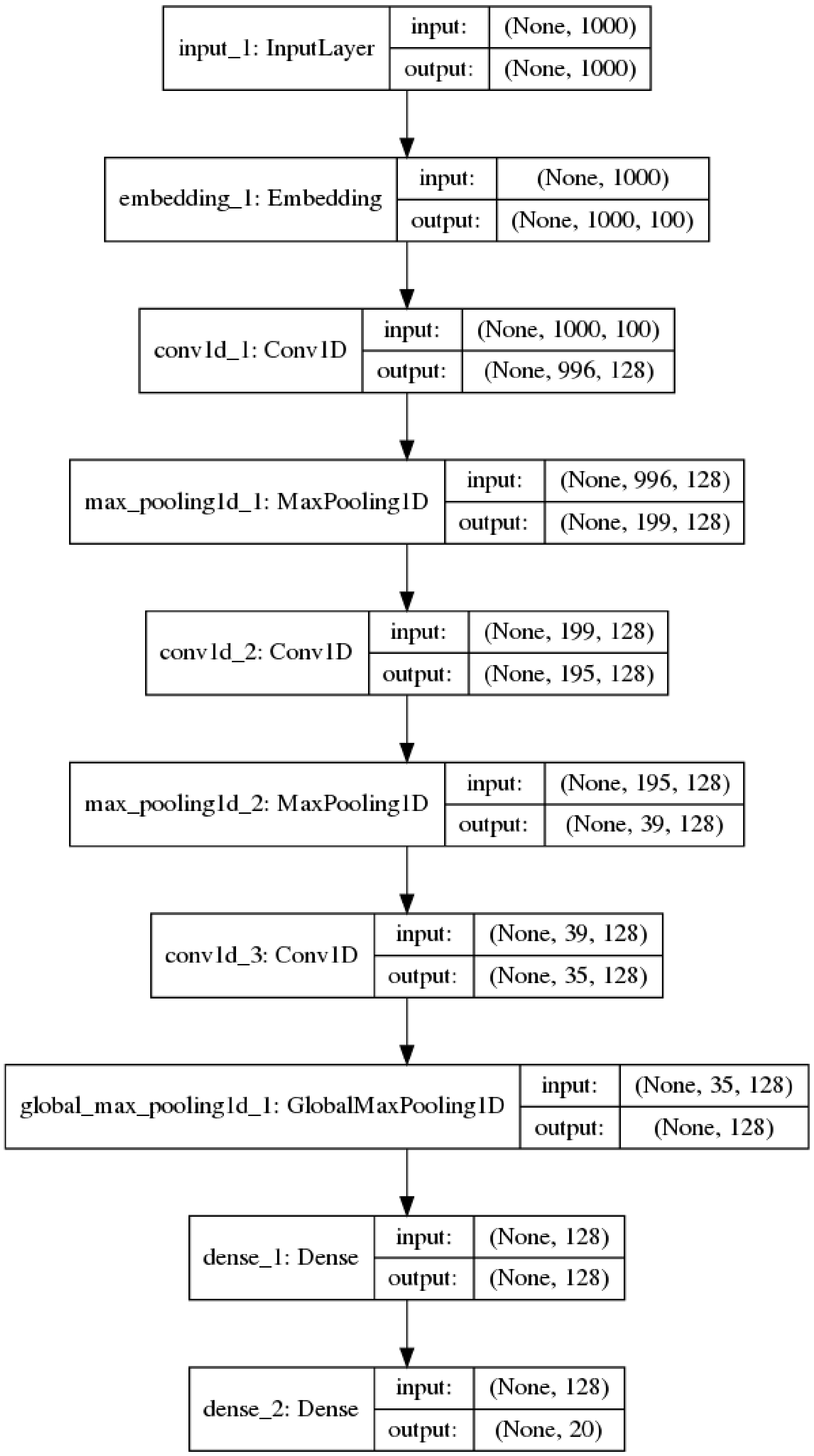
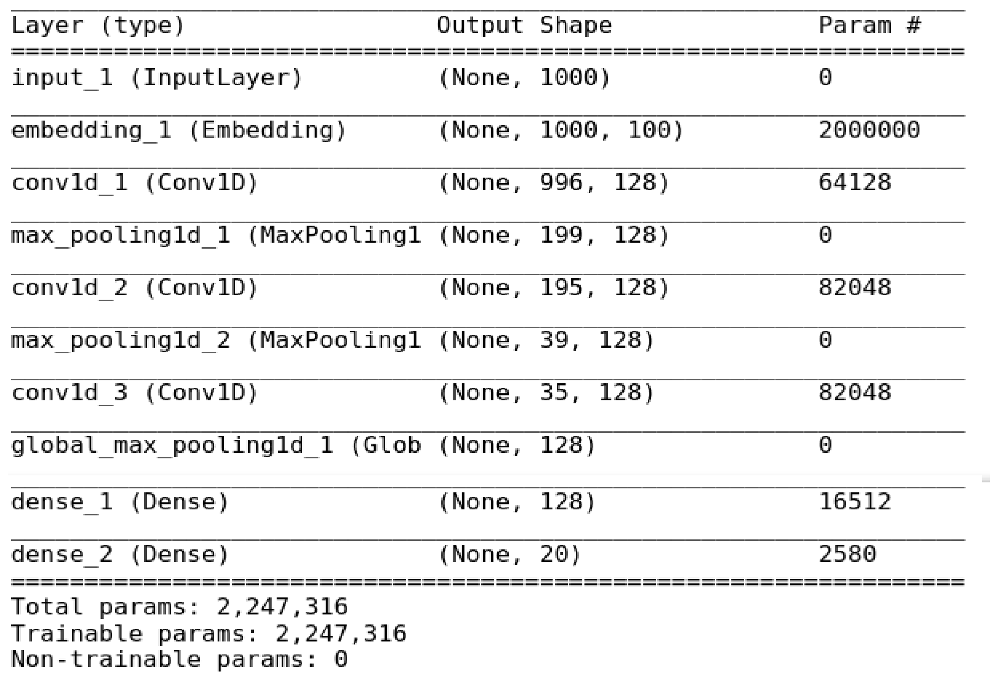
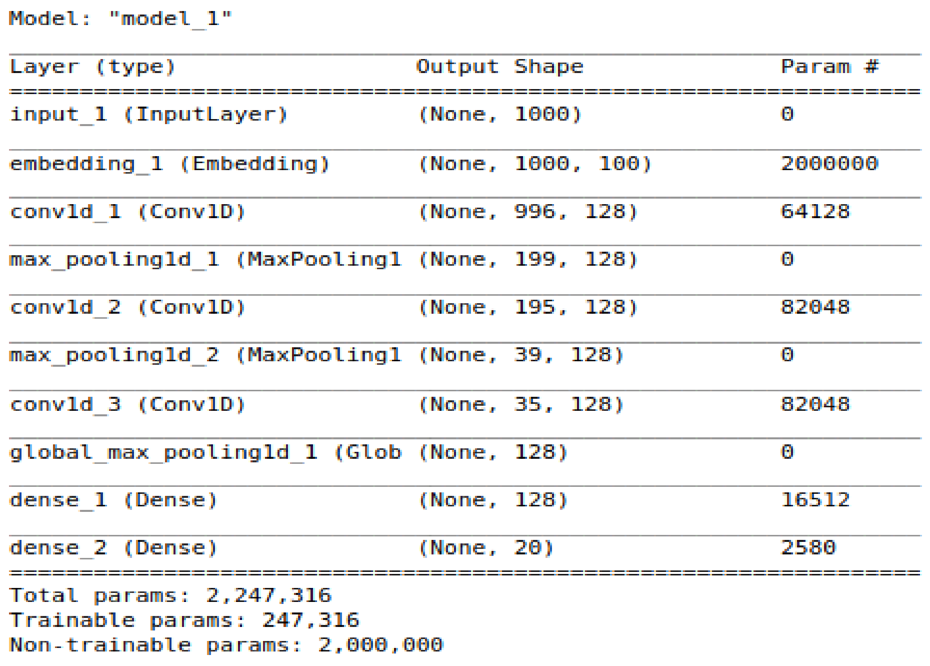


Figure 4.6: Architecture of proposed CNN



```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 1000)              0
_________________________________________________________________
embedding_1 (Embedding)      (None, 1000, 100)         2000000
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 996, 128)          64128
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 199, 128)          0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 195, 128)          82048
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 39, 128)           0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 35, 128)           82048
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_2 (Dense)              (None, 20)                2580
=================================================================
Total params: 2,247,316
Trainable params: 2,247,316
Non-trainable params: 0
```

Figure 4.7: Model summary with trainable true.



```
Model: "model_1"

Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 1000)              0
_________________________________________________________________
embedding_1 (Embedding)      (None, 1000, 100)         2000000
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 996, 128)          64128
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 199, 128)          0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 195, 128)          82048
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 39, 128)           0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 35, 128)           82048
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_2 (Dense)              (None, 20)                2580
=================================================================
Total params: 2,247,316
Trainable params: 247,316
Non-trainable params: 2,000,000
```

Figure 4.8: Model summary with trainable false

- **CNN Hyper-parameters Optimization**

There are many parameters to tweak and choose from, those parameter are called hyper-parameters. A parameter sweeping used to performing for some of hyper parameters optimization by conducting a lot of tweaking and experimenting until find the best optimization.

1. Pre-trained Word Embedding

The text represented as a vector in Naïve Bayes, without consider the similarity of the words. In CNN the pre-trained Glove, word embedding used, with considering the similarity of words instead of looking for a single word. Enabling the embedding to be update during training by set trainable parameter =True, will positively affect the classifier accuracy.

2. Batch Size

The batch size is a hyper parameter that defines the number of samples to work through before updating the internal model parameters, it’s like a for-loop iterating over one or more samples until propagated all samples through of the network and making predictions. At the end of the batch, the predictions compared to the expected output variables and an error is calculated.

There are no magic rules for how to configure these parameters. Just keep trying with different values to get the best for the problem. Batch size is the number of training examples in one forward/backward pass. The higher the batch size, the more memory space you need. The texting results show that using a larger batch there is a degradation in the quality of the model, as measured by its ability to generalize.

3. Epochs

The number of epochs is a hyper-parameter that defines the number times that the learning algorithm will work through the entire training dataset. It is the number of times the algorithm will train.

4. Activation Functions

Activation functions is used to introduce nonlinearity to models. Various activation functions can be used with CNN. It is generally common to use a rectified linear unit (ReLU) for hidden layers, The function is attached to each neuron in the network, and determines whether it should be activated or not, based on whether neuron's input is relevant for the model prediction or not. A sigmoid function used for the output layer in a binary classification problem, a softmax function for the output layer of multi-class classification problems. Softmax able to handle multiple classes normalizes the outputs for each class between zero and one, and divides by their sum, giving the probability of the input value being in a specific class. Typically Softmax equation (4.8) is used only for the output layer, to classify inputs into multiple categories.

$$f_i(\vec{a}) = \frac{e^{a_i}}{\sum_k e^{a_k}}$$

Softmax Activation function)

(4.8)

5. Optimizer

Optimizer is an algorithm used in CNN classification model to update weights of every layer after every iteration. The error determined by a loss function, loss minimize by using optimizer.

6. Convolutional Layer

Convolutional is mathematical combination of two relation to produce third relation. The convolutional is passed over layer of input neurons multiplied by a weight to extract features, which generate multiple feature map. The Rectified Linear Unit (ReLU) activation function is moved over the output to produce a nonlinear relationship for the output.

7. Max Pooling Layer

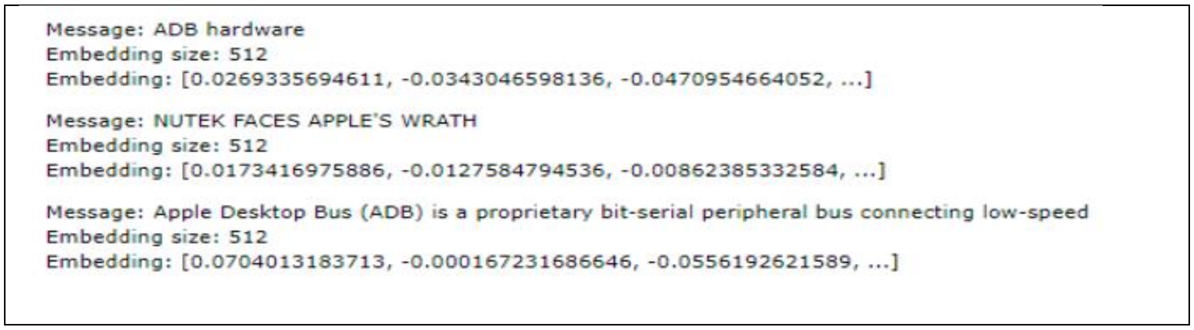
The max pooling layer applied between the convolutional layers, to reduce the high computation of CNN and make sure not to over fit data, by keeping the significant information of the convolutional and sent it to the next layer.

4.5.3. Similarity Check Experiment

Python programming language used to implement this model. With help of universal sentence encoder. The model trained and optimize for different text length such as word, sentences, and paragraph. The input is variable length English text, and the output is a 512 dimensional vector, represent a word, sentence or paragraph.

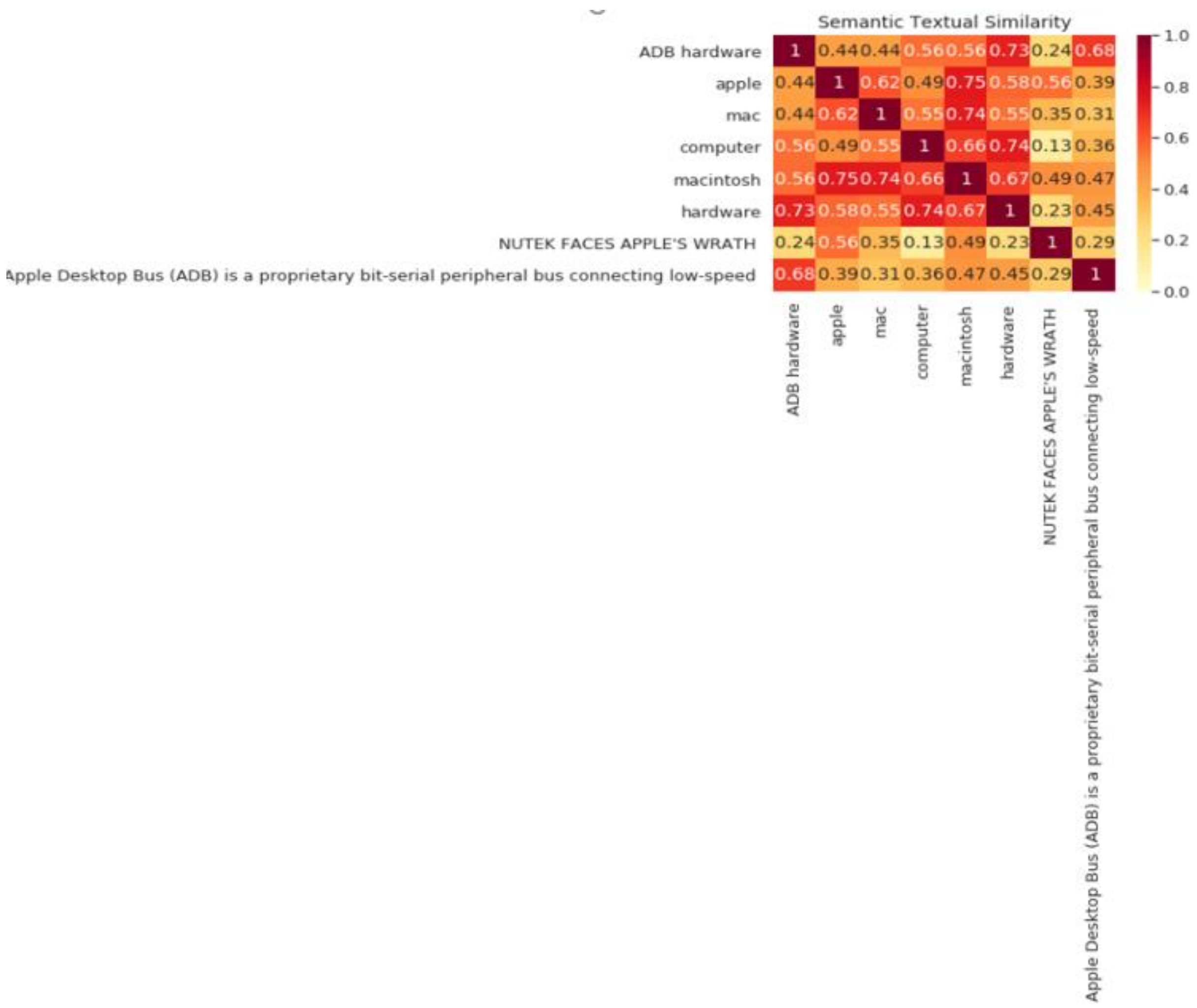
Table. (4.2) shows the result of the 512 dimensional vector for, word: "Apple", sentence:" NUTEK FACES APPLE'S WRATH", paragraph: "Apple Desktop Bus (ADB) is a proprietary bit-serial peripheral bus connecting low-speed".

Table 4.2: 512 Dimensional vector



```
Message: ADB hardware
Embedding size: 512
Embedding: [0.0269335694611, -0.0343046598136, -0.0470954664052, ...]

Message: NUTEK FACES APPLE'S WRATH
Embedding size: 512
Embedding: [0.0173416975886, -0.0127584794536, -0.00862385332584, ...]

Message: Apple Desktop Bus (ADB) is a proprietary bit-serial peripheral bus connecting low-speed
Embedding size: 512
Embedding: [0.0704013183713, -0.000167231686646, -0.0556192621589, ...]
```

Fig. (4.9) show the results of the similarity in a heat map. The final graph is a matrix where each entry [i, j] is colored based on the inner product of the encodings for sentence i and j.



	ADB hardware	apple	mac	computer	macintosh	hardware	NUTEK FACES APPLE'S WRATH	Apple Desktop Bus (ADB) is a proprietary bit-serial peripheral bus connecting low-speed
ADB hardware	1	0.44	0.44	0.56	0.56	0.73	0.24	0.68
apple	0.44	1	0.62	0.49	0.75	0.58	0.56	0.39
mac	0.44	0.62	1	0.55	0.74	0.55	0.35	0.31
computer	0.56	0.49	0.55	1	0.66	0.74	0.13	0.36
macintosh	0.56	0.75	0.74	0.66	1	0.67	0.49	0.47
hardware	0.73	0.58	0.55	0.74	0.67	1	0.23	0.45
NUTEK FACES APPLE'S WRATH	0.24	0.56	0.35	0.13	0.49	0.23	1	0.29
Apple Desktop Bus (ADB) is a proprietary bit-serial peripheral bus connecting low-speed	0.68	0.39	0.31	0.36	0.47	0.45	0.29	1

Figure 4.9: A prototype of words and sentence similarity check.

4.6 Evaluation Measures

4.6.1. MNB Model Evaluation

A series of experiments were conducted on Multinomial Naïve Bayes (MNB) by using a 20-newsgroup dataset, to study the performance of the MNB model. It experimented with BOW and TF-IDF methods for features extraction. Table (4.2) show default parameters value set for the MNB classifiers.

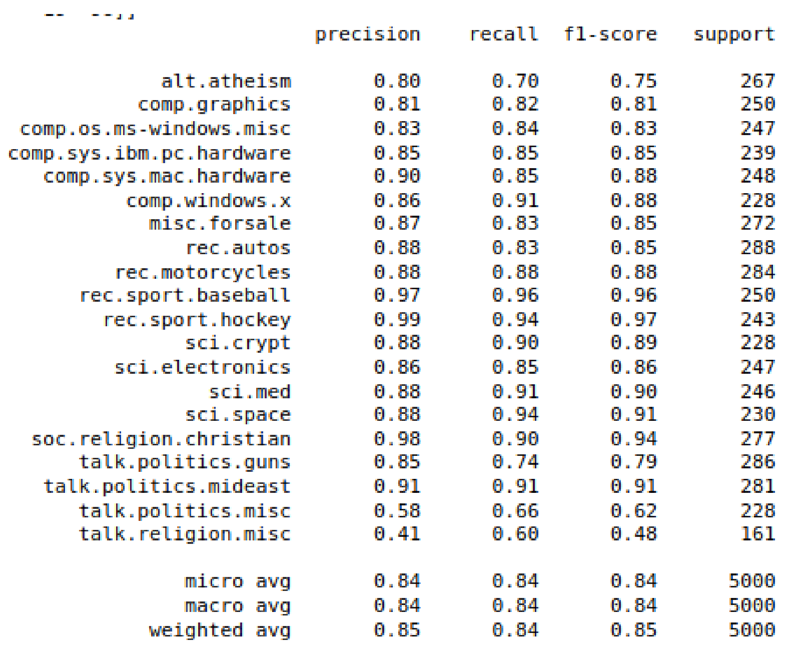
Table 4.3: MNB parameters

Parameter	Value
Validation Split	.25%
alpha	1.0
class_prior	None
fit_prior	True

Experiment 1:

To study the performance of the proposed mechanism, several evaluation metrics were calculated. The following four evaluation matrices are commonly used to evaluate MNB algorithms: classification accuracy, recall, precision, and f1_score.

Fig. (4.10) show the result of evaluation matrices for Multinomial Naïve Bayes classifier with BOW. The achieved accuracy is 0.8426%.

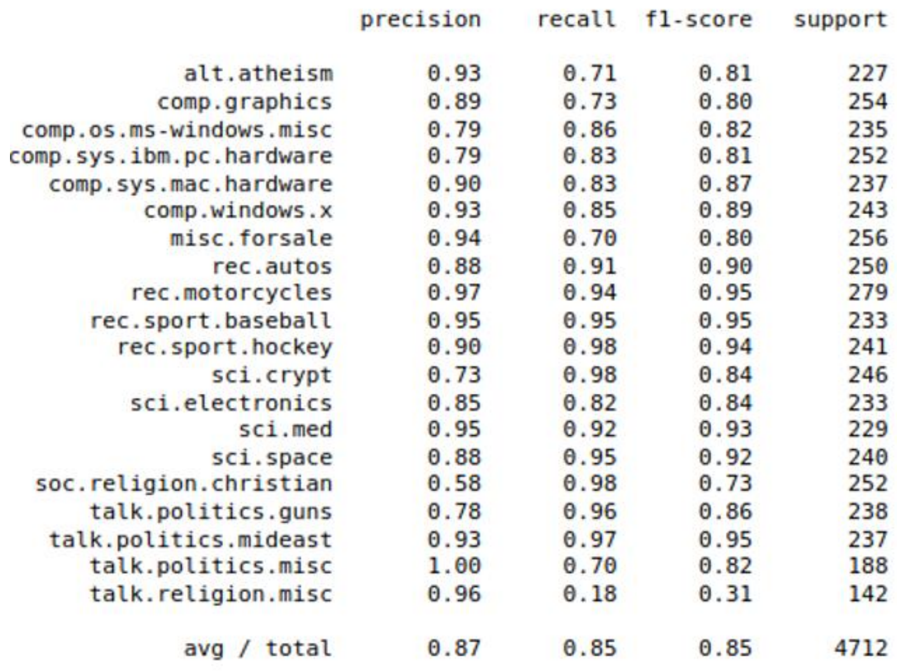


```
                          precision    recall  f1-score   support

             alt.atheism       0.80      0.70      0.75       267
           comp.graphics       0.81      0.82      0.81       250
 comp.os.ms-windows.misc       0.83      0.84      0.83       247
comp.sys.ibm.pc.hardware       0.85      0.85      0.85       239
   comp.sys.mac.hardware       0.90      0.85      0.88       248
          comp.windows.x       0.86      0.91      0.88       228
            misc.forsale       0.87      0.83      0.85       272
               rec.autos       0.88      0.83      0.85       288
         rec.motorcycles       0.88      0.88      0.88       284
      rec.sport.baseball       0.97      0.96      0.96       250
        rec.sport.hockey       0.99      0.94      0.97       243
               sci.crypt       0.88      0.90      0.89       228
         sci.electronics       0.86      0.85      0.86       247
                 sci.med       0.88      0.91      0.90       246
               sci.space       0.88      0.94      0.91       230
  soc.religion.christian       0.98      0.90      0.94       277
      talk.politics.guns       0.85      0.74      0.79       286
   talk.politics.mideast       0.91      0.91      0.91       281
      talk.politics.misc       0.58      0.66      0.62       228
      talk.religion.misc       0.41      0.60      0.48       161

               micro avg       0.84      0.84      0.84      5000
               macro avg       0.84      0.84      0.84      5000
            weighted avg       0.85      0.84      0.85      5000
```

Figure 4.10: Result of evaluation matrices for Multinomial Naïve Bayes classifier with BOW

Experiment 2:

Fig. (4.11) shows the result of evaluation matrices for Multinomial Naïve Bayes classifier with TF-IDF. The achieved accuracy is 0.8531%.



```
                          precision    recall  f1-score   support

             alt.atheism       0.93      0.71      0.81       227
           comp.graphics       0.89      0.73      0.80       254
 comp.os.ms-windows.misc       0.79      0.86      0.82       235
comp.sys.ibm.pc.hardware       0.79      0.83      0.81       252
   comp.sys.mac.hardware       0.90      0.83      0.87       237
          comp.windows.x       0.93      0.85      0.89       243
            misc.forsale       0.94      0.70      0.80       256
               rec.autos       0.88      0.91      0.90       250
         rec.motorcycles       0.97      0.94      0.95       279
      rec.sport.baseball       0.95      0.95      0.95       233
        rec.sport.hockey       0.90      0.98      0.94       241
               sci.crypt       0.73      0.98      0.84       246
         sci.electronics       0.85      0.82      0.84       233
                 sci.med       0.95      0.92      0.93       229
               sci.space       0.88      0.95      0.92       240
  soc.religion.christian       0.58      0.98      0.73       252
      talk.politics.guns       0.78      0.96      0.86       238
   talk.politics.mideast       0.93      0.97      0.95       237
      talk.politics.misc       1.00      0.70      0.82       188
      talk.religion.misc       0.96      0.18      0.31       142

             avg / total       0.87      0.85      0.85      4712
```

Figure 4.11: Result of evaluation matrices for Multinomial Naïve Bayes classifier with TF-IDF

4.6.2. CNN Model Evaluation

A series of experiments were conducted on convolution neural network (CNN) by using a 20-newsgroup dataset, to study the performance of the CNN model. It experimented with model parameters to check the accuracy by increasing the number of epochs, batch size, and used different optimizers. Table (4.4) shows default parameters value set for the CNN classifiers.

Table 4.4: Default CNN parameters

Parameter Name	Value
Max number of words	1000
Embedding Dimension	100
Validation Split	.25%
Activation function	softmax
Word Embedding	Glove

To study the performance of the CNN model, accuracy evaluation metrics were calculated. Table (4.5) summarize the accuracy results of using different optimizer in CNN classifier.

Table 4.5: Tuning CNN hyper-parameter

Experiment number.	Number of epochs	Number of batch size	optimizer	set trainable parameters	Training Accuracy	Testing Accuracy
Training ration 75% testing .25%						
1	10	128	rmsprop	True	0.9651	0.7958
2	10	128	rmsprop	False	0.9154	0.7069
3	15	128	rmsprop	True	0.9694	0.7916
4	15	256	rmsprop	True	0.9676	0.7970
5	15	256	Adam	True	0.9587	0.7928
6	15	256	sgd	True	0.4701	0.4367
7	100	128	rmsprop	True	0.9622	0.7918

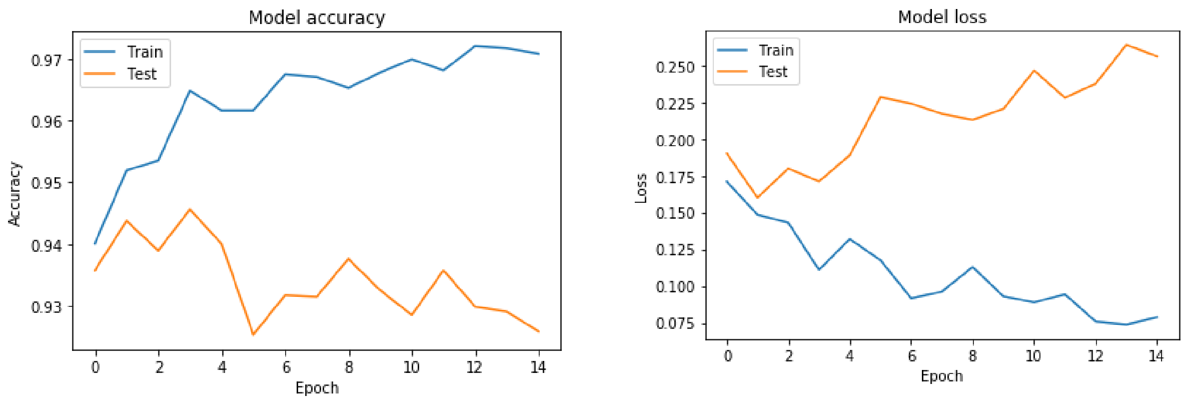


Figure 4.12: Best accuracy and model loss for CNN classifier

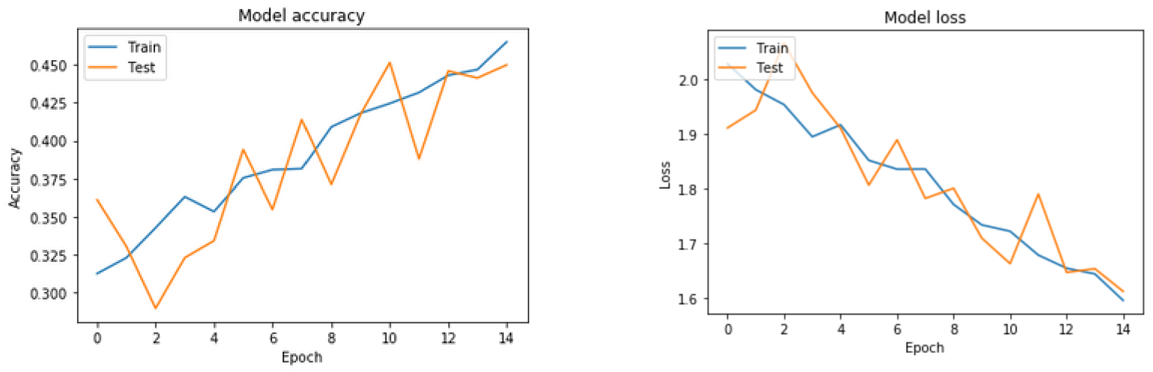


Figure 4.13: Worst accuracy and model loss for CNN classifier

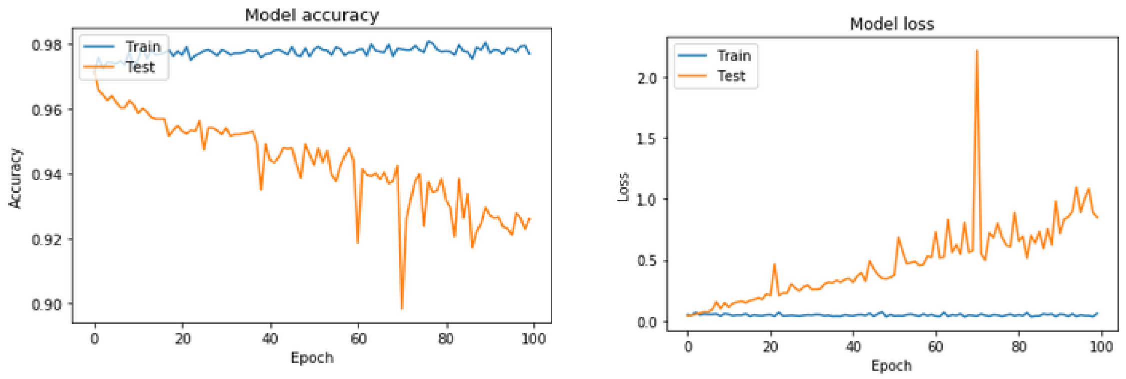


Figure 4.14: 100 Epoch with 256 batch size

Table (4.6) shows the results of using different batch size on 10, and 15 epoch.

Table 4.6: Batch size comparison

Experiment Number	Batch Size	512	256	128	64
Training ratio 75% testing .25% Trainable True 15 Epoch					
1	Training Accuracy	0.9555	0.9661	0.9652	0.9643
	Testing Accuracy	0.7782	0.7902	0.7862	0.7828
Training ratio 75% testing .25% Trainable True 10 Epoch					
2	Training Accuracy	0.9303	0.8036	0.9651	0.9616
	Testing Accuracy	0.7842	0.6675	0.7958	0.7908

Referring to Table 4.5 and Table 4.6 we conclude the following :

1. The CNN has many hyper-parameters and it is not easy to optimize.
2. The batch size and the optimizer have a high impact on the CNN model, the results shows that using stochastic gradient descent (SGD) optimizer with a high learning rate will reduce the model accuracy. However, the (rmsprop) optimizer is the most suitable optimizers, Fig. (4.12) and Fig. (4.13) show the result of using (SGD) and (RMSprop) optimizers.
3. The testing results shows that the after 15 epochs, the accuracy was not enhanced. Fig. (4.14) shows the result of the CNN model accuracy with 100 epoch.

Table 4.7: Evaluation results of classification model

Model	Accuracy
Multinomial Naïve Bayes with BOW	0.8426%
Multinomial Naïve Bayes with TF-IDF	0.853%
CNN with (rmsprop) optimizer	0.7970%
CNN with (sgd) optimizer	0.4367

Referring to Table (4.7), we conclude the following :

1. All the considered models achieved high and acceptable results for all evaluation metrics in text classification excepts for the CNN model with stochastic gradient descent (SGD). The SGD optimizer performs computation in small subset instead of performing computation in the whole set , SGD used low learning rate that make the learning slow.
2. Multinomial Naïve Bayes with TF-IDF performs better.

4.6.3. Similarity Check and Ranking Model Evaluation

A series of experiments are conducted on similarity check model, using a sample text and documents from 20-newsgroup dataset, to study the performance of this model, experimented with different word length to test the accuracy of model.

The cos similarity formula used to compute ditance between the keyword and its synonyms based on Glove for pre-trained word embedding to compare two vectors for how similar their direction are (Cer, et al. 2018). The BM25 formula used to compute the ranking score for display results with search query with existing documents in corpus (Garcia 2016), Table (4.8) show default parameters value set for the MB25.

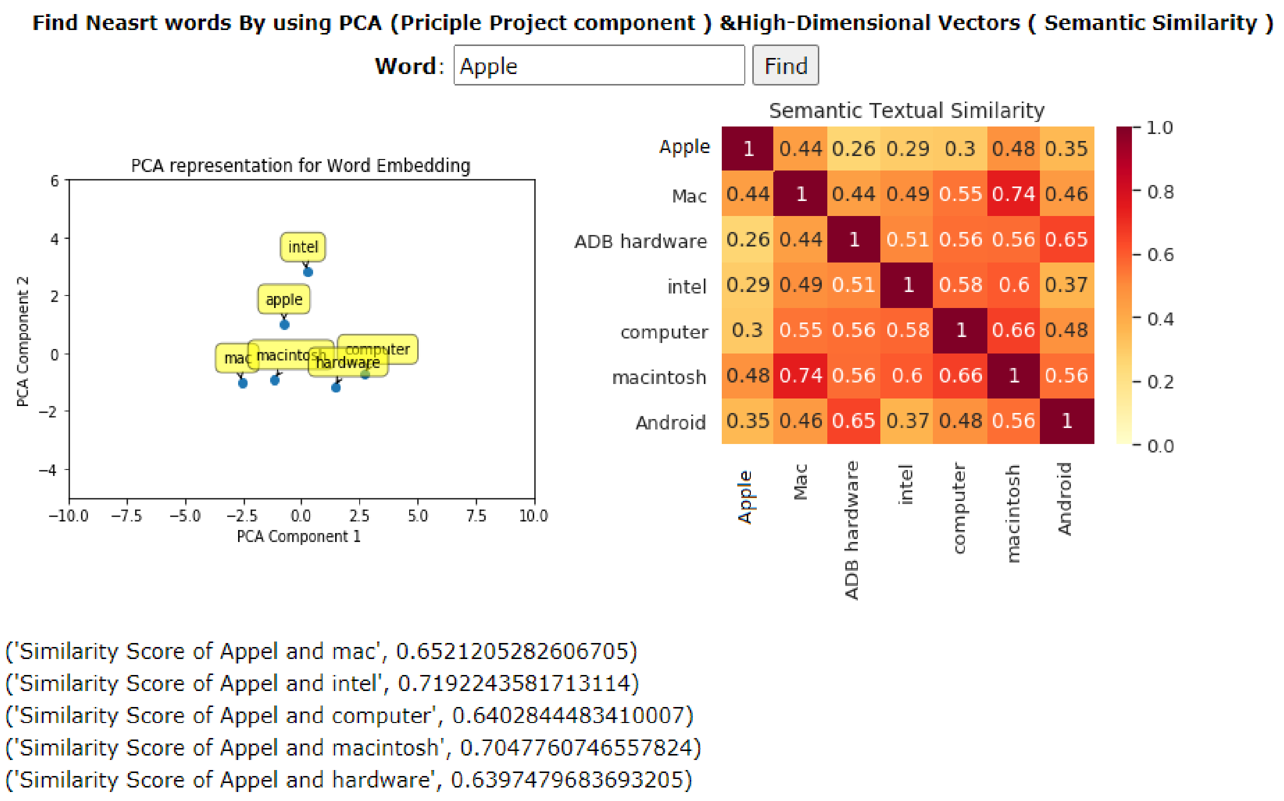
Table 4.8: BM25 default parameters

Parameter	Value
k1 : float	1.5
b : float	0.75

Experment 1:

For the explanation purposes the model will test the "Apple" keyword and finding its synonyms. The result show that "intel, mac, macintosh, hardware, and computer" are close words for the keyword "Apple" based on computing the cosine similaity equation.

Fig.(4.15) show a results of words similarity check model in Al-Quds system. it find the nearest word by using PCA in high-dimensional vectors and cosine similarity for (apple) keyword, by compute the representation of each sentences in fixed dimension length and compute the cosine distance between these vectors.


Find Neasrt words By using PCA (Priciple Project component) &High-Dimensional Vectors (Semantic Similarity)
Word: Apple
Find
PCA representation for Word Embedding
PCA Component 1
PCA Component 2
intel
apple
mac
macintosh
computer
hardware
Semantic Textual Similarity

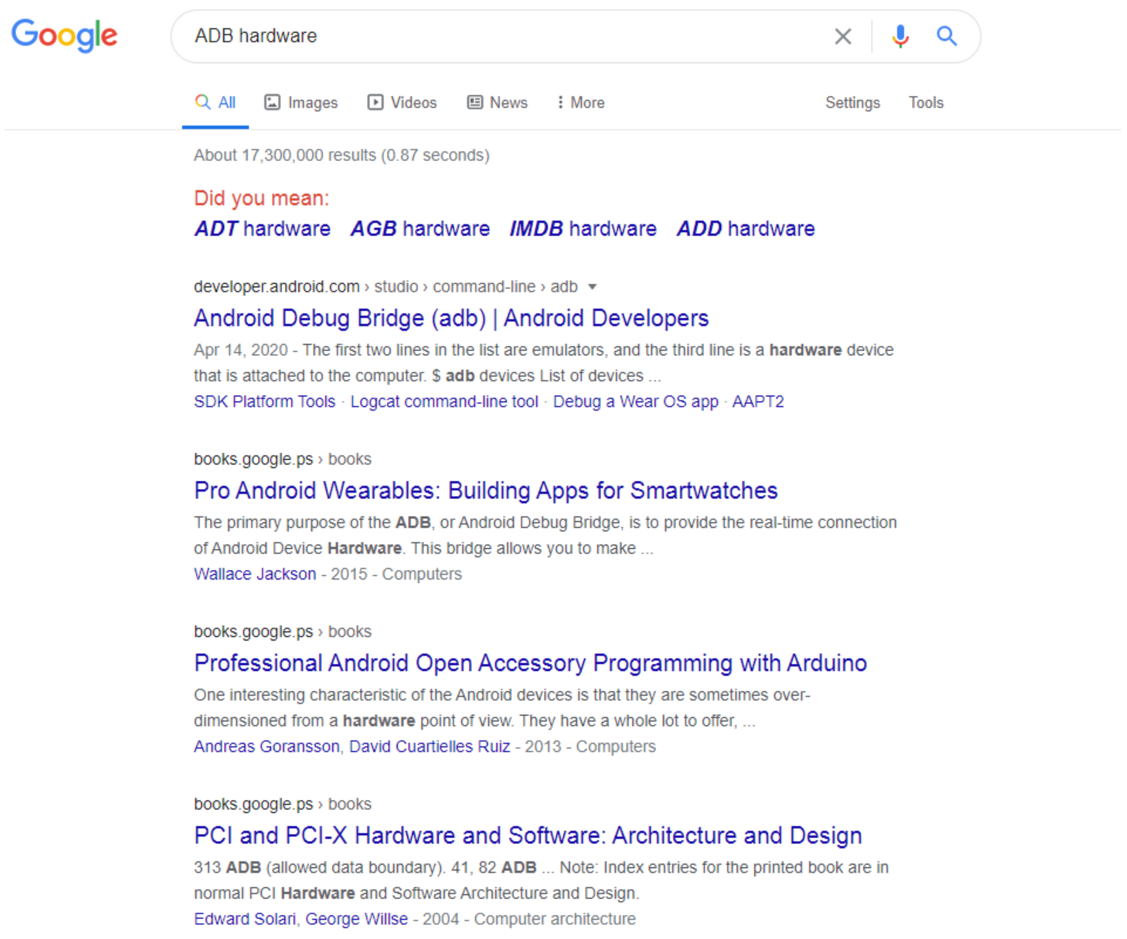
	Apple	Mac	ADB hardware	intel	computer	macintosh	Android
Apple	1	0.44	0.26	0.29	0.3	0.48	0.35
Mac	0.44	1	0.44	0.49	0.55	0.74	0.46
ADB hardware	0.26	0.44	1	0.51	0.56	0.56	0.65
intel	0.29	0.49	0.51	1	0.58	0.6	0.37
computer	0.3	0.55	0.56	0.58	1	0.66	0.48
macintosh	0.48	0.74	0.56	0.6	0.66	1	0.56
Android	0.35	0.46	0.65	0.37	0.48	0.56	1

```
('Similarity Score of Appel and mac', 0.6521205282606705)
('Similarity Score of Appel and intel', 0.7192243581713114)
('Similarity Score of Appel and computer', 0.6402844483410007)
('Similarity Score of Appel and macintosh', 0.7047760746557824)
('Similarity Score of Appel and hardware', 0.6397479683693205)
```

Figure 4.15: Results of word similarity

Experiment 2:

A sample query used to compare the result of the Al-Quds system with a search engine such as Google. For instance, if google used to search for ADB hardware, the results almost show links related to android as shown screenshot Fig (4.16), likewise, searching for "ADB hardware" from a corpus shown in Table (4.9) which four documents from the same classification categories in the corpus


ADB hardware

All Images Videos News More Settings Tools

About 17,300,000 results (0.87 seconds)

Did you mean:
ADT hardware *AGB* hardware *IMDB* hardware *ADD* hardware

developer.android.com › studio › command-line › adb

Android Debug Bridge (adb) | Android Developers

Apr 14, 2020 - The first two lines in the list are emulators, and the third line is a **hardware** device that is attached to the computer. $ **adb** devices List of devices ...

SDK Platform Tools · Logcat command-line tool · Debug a Wear OS app · AAPT2

books.google.ps › books

Pro Android Wearables: Building Apps for Smartwatches

The primary purpose of the **ADB**, or Android Debug Bridge, is to provide the real-time connection of Android Device **Hardware**. This bridge allows you to make ...

Wallace Jackson - 2015 - Computers

books.google.ps › books

Professional Android Open Accessory Programming with Arduino

One interesting characteristic of the Android devices is that they are sometimes over-dimensioned from a **hardware** point of view. They have a whole lot to offer, ...

Andreas Goransson, David Cuartielles Ruiz - 2013 - Computers

books.google.ps › books

PCI and PCI-X Hardware and Software: Architecture and Design

313 **ADB** (allowed data boundary). 41, 82 **ADB** ... Note: Index entries for the printed book are in normal PCI **Hardware** and Software Architecture and Design.

Edward Solari, George Willse - 2004 - Computer architecture

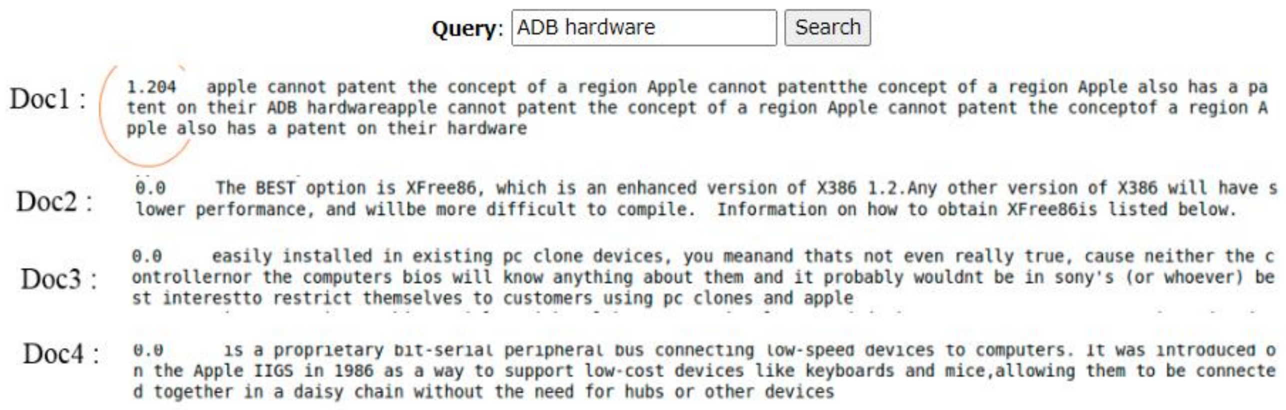
Figure 4.16: Screenshot for google search results

Table 4.9: Test documents from a corpus

Doc1	Apple cannot patent the concept of a region Apple cannot patent the concept of a region Apple also has a patent on their ADB hardware apple cannot patent the concept of a region Apple cannot patent the concept of a region Apple also has a patent on their hardware"
Doc2	The BEST option is XFree86, which is an enhanced version of X386 1.2 Any other version of X386 will have slower performance, and will be more difficult to compile. Information on how to obtain XFree86 is listed below.
Doc3	Easily installed in existing pc clone devices, you mean and that’s not even really true, cause neither the controller. nor the computers bios will know anything about them" and it probably wouldn’t be in sony's (or whoever) best interest to restrict themselves to customers using pc clones and apple
Doc4	comp.sys.mac.hardware Appel, Mac, computer, Macintosh ADB hardware " is a proprietary bit-serial peripheral bus connecting low-speed devices to computers."It was introduced on the Apple IIGS in 1986 as a way to support low-cost devices like keyboards and mice,"allowing them to be connected together in a daisy chain without the need for hubs or other devices

Source of text (Home Page for 20 Newsgroups Data Set., n.d)

Fig. (4.17) shows the search result, where the result displayed only Doc1 as a relevant document. By using Al-Quds System for the same corpus and same search term, the search result displayed more related documents, as shown in Fig (4.18).



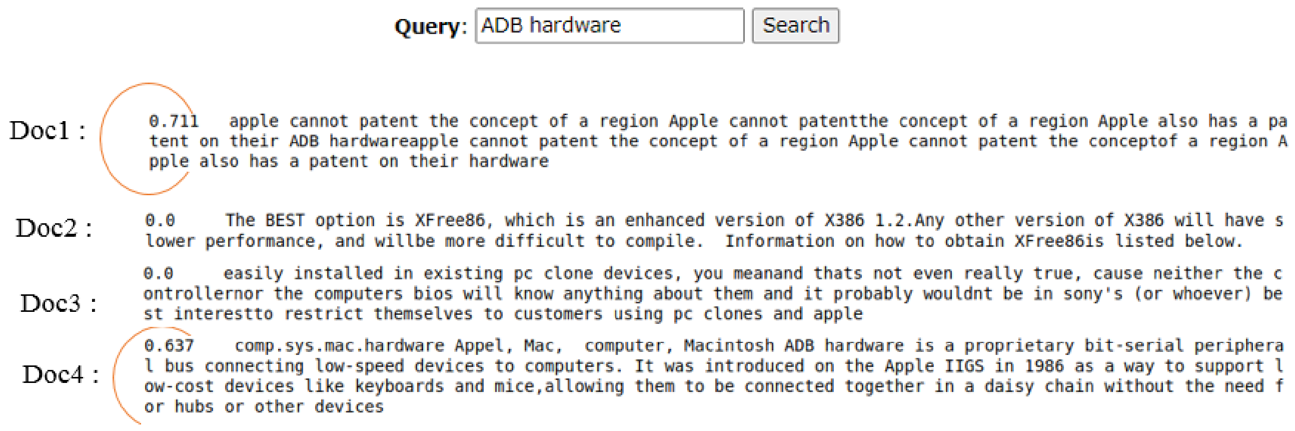
Doc1 : 1.204 apple cannot patent the concept of a region Apple cannot patentthe concept of a region Apple also has a patent on their ADB hardwareapple cannot patent the concept of a region Apple cannot patent the conceptof a region Apple also has a patent on their hardware

Doc2 : 0.0 The BEST option is XFree86, which is an enhanced version of X386 1.2.Any other version of X386 will have slower performance, and willbe more difficult to compile. Information on how to obtain XFree86is listed below.

Doc3 : 0.0 easily installed in existing pc clone devices, you meanand thats not even really true, cause neither the controllernor the computers bios will know anything about them and it probably wouldnt be in sony's (or whoever) best interestto restrict themselves to customers using pc clones and apple

Doc4 : 0.0 is a proprietary bit-serial peripheral bus connecting low-speed devices to computers. It was introduced on the Apple IIGS in 1986 as a way to support low-cost devices like keyboards and mice,allowing them to be connected together in a daisy chain without the need for hubs or other devices

Figure 4.17: Screenshot for search results



Doc1 : 0.711 apple cannot patent the concept of a region Apple cannot patentthe concept of a region Apple also has a patent on their ADB hardwareapple cannot patent the concept of a region Apple cannot patent the conceptof a region Apple also has a patent on their hardware

Doc2 : 0.0 The BEST option is XFree86, which is an enhanced version of X386 1.2.Any other version of X386 will have slower performance, and willbe more difficult to compile. Information on how to obtain XFree86is listed below.

Doc3 : 0.0 easily installed in existing pc clone devices, you meanand thats not even really true, cause neither the controllernor the computers bios will know anything about them and it probably wouldnt be in sony's (or whoever) best interestto restrict themselves to customers using pc clones and apple

Doc4 : 0.637 comp.sys.mac.hardware Appel, Mac, computer, Macintosh ADB hardware is a proprietary bit-serial peripheral bus connecting low-speed devices to computers. It was introduced on the Apple IIGS in 1986 as a way to support low-cost devices like keyboards and mice,allowing them to be connected together in a daisy chain without the need for hubs or other devices

Figure 4.18: Search results by using Al-Quds System

Referring to the results of experiments 1, 2, we conclude that the considered methods achieve high and acceptable results. The results proved the efficiency of Al-Quds System to improve search result relevance and presentation by displaying more relevant results with its ranking score. Additional scenario can found in appendix 1.

From the experiments, we conclude that using a MNB for text and document classification will give superior results, and it’s a good classifier to use if the semantic of words not an issue, and the classifiers accuracy depends primarily on feature extraction and dimensionality reduction, In addition, the result shows that using NLP transfer learning and cosine similarity to compute the words similarity will produce high accuracy results for word and sentences similarity.

Chapter5

Conclusion and Future Works

This chapter summarizes the main conclusions and highlights some of the future works for further improvement.

5.1 Conclusion

This thesis mainly aims at proposing frameworks based on machine learning and deep learning concepts to identify an efficient and effective way to improve search results without overburdening the authors or searchers. This is done by increasing the relevancy of scholar search results based on quantitative and qualitative analysis of text classification, similarity check, and metadata.

The results proved the efficiency of the proposed framework in the scholar search relevance optimization problem. The classifier model gives the ability for text and document classification, and the words similarity gives the ability to recommend synonyms keyword base on publication subject or description. Moreover, it computes the similarity between subject and description keywords, where the closer the cosine value to 1, the smaller the angle, and the higher match between words vectors. The metadata model gives the ability to utilize metadata capabilities to optimize some of SEO parameters, especially the keywords and content description by enabling authors to add as much relevant information as possible to enhance the relevancy between the search query and display results, and increase the ranking score of publications

The results have shown that the MNB and CNN are suitable techniques to use for classification problems. A simple classifier such as MNB gave 85% accuracy results for the classification task where the CNN gave 79% accuracy on the used dataset, in case the vocabulary or features are not changed, in which it's represents a text as vectors and runs algorithm on these vectors. This means if the text changes the classifier needs to represent the new text again to extract new features for the classifier, while CNN considers the similarity of words in a different review, which will help to look at the review as whole instead of focusing on every single word. The word embedding is responsible to convert text into numeric vectors such as Glove. This makes it possible to compute the distance between words and find the relationship between words based on its represented vectors in the vector space. The generalization of vector representation is the fundamental part of a certain effective deep neural network model for NLP. Customizing the classifier's hyper-parameters can dramatically affect the classifier's accuracy and performance. In addition, the results show that using the DAN is suitable for a similarity check. Moreover, the results indicate using a transfer learning in natural langue processing tasks saves time and effort, especially when working with words embedding and use a relatively simple model that uses the capabilities of modern NLP without a lot of unnecessary complexity, just as the deep averaging network. Therefore, combining language processing, deep learning and metadata enhances search results by increasing the relevancy between display result and search query.

5.2 Future Work

Even though this study provides an answer to the research questions, several limitations and possible improvements are noticed. For future works, the following research lines can be followed to improve the search relevance:

1. Create Meta2Vec model like word2vec and Doc2vec, to create a Meta-Embedding matrix to use with CNN.
2. The extra improvement could be applied to Al-Quds System, such as auto-fill and extract of metadata attributes, make integration with search engines, allows user to evaluate the search results.
3. Use different datasets than 20 newsgroups, especially the Arabic datasets.
4. Extend the proposed framework to work with Arabic word embedding.
5. Add autofill for metadata fields.
6. Add more optimization options such as citation.

The End

• References

Abbas, Muhammad, Saleem Memon, Kamran Ali, and Abdul Jamali. 2019. “Multinomial Naive Bayes Classification Model for Sentiment Analysis.” *IJCSNS International Journal of Computer Science and Network Security* 19 (3): 62. https://doi.org/10.13140/RG.2.2.30021.40169.

Akritidis, Leonidas, and Panayiotis Bozanis. 2013. “A Supervised Machine Learning Classification Algorithm for Research Articles.” *Proceedings of the ACM Symposium on Applied Computing*, no. June 2019: 115–20. https://doi.org/10.1145/2480362.2480388.

Amin, Muhammad Zain, and Noman Nadeem. 2018. “Convolutional Neural Network: Text Classification Model for Open Domain Question Answering System,” no. September. http://arxiv.org/abs/1809.02479.

Anurag Sarkar, Saptarshi Chatterjee, Writayan Das, Debabrata Datta. 2015. “Text Classification Using Support Vector Machine Anurag.” *International Journal of Engineering Science Invention* 8 (2): 33–37.

Awasthi, Shipra, and Babita Jaiswal. 2008. “Open Archives Metadata Harvesting : An Overview.” *6th Convention PLANNER-2008, Nagaland University, Nagaland, Nov 06-07, 2008*, 4–10.

Azimjonov, Jahongir, and Jumabek Alikhanov. 2018. “Rule Based Metadata Extraction Framework from Academic Articles.”

Barker, Phil, and Lorna M Campbell. 2010. “Metadata for Learning Materials: An Overview of Existing Standards and Current Developments.” *Cognition and Learning* 7: 225–43. http://www.oldcitypublishing.com/TICL/TICLcontents/TICLv7n3-4contents.html.

Beel, Jöran, Bela Gipp, and Erik Wilde. 2010. “Academic Search Engine Optimization (<span Style="font-Variant: Small-Caps">ASEO/Span).” *Journal of Scholarly Publishing* 41 (2): 176–90. https://doi.org/10.3138/jsp.41.2.176.

Cer, Daniel, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, et al. 2018. “Universal Sentence Encoder.” *EMNLP 2018 - Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Proceedings*, 169–74. http://arxiv.org/abs/1803.11175.

Cer, Daniel, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, et al. 2018. “Universal Sentence Encoder for English.” *EMNLP 2018 - Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Proceedings*, 169–74. https://doi.org/10.18653/v1/d18-2029.

Chiticariu, Laura, Yunyao Li, and Frederick R. Reiss. 2013. “Rule-Based Information Extraction Is Dead! Long Live Rule-Based Information Extraction Systems!” *EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the*

Conference, no. October: 827–32.

Cock, M. De, S. Guadarrama, Martine De Cock, M. Nikravesh, Sergio Guadarrama, and Masoud Nikravesh. 2005. *Soft Computing for Information Processing and Analysis*. *Soft Computing for Information Processing and Analysis*. Vol. 164. https://doi.org/10.1007/3-540-32365-1.

"Confusion Matric(TPR,FPR,FNR,TNR), Precision, Recall, F1-Score | by Namratesh Shrivastav | Data Driven Investor | Medium." n.d. Accessed July 21, 2020. https://medium.com/datadriveninvestor/confusion-matric-tpr-fpr-fnr-tnr-precision-recall-f1-score-73efa162a25f.

"DC -- Template." n.d. Accessed August 11, 2020. http://metadataetc.org/dctemplate.html.

"DCMI: Dublin Core^{TM} Metadata Element Set, Version 1.1: Reference Description." n.d. Accessed May 10, 2020. https://www.dublincore.org/specifications/dublin-core/dces/.

Deng, Sai, and Terry Reese. 2009. "Customized Mapping and Metadata Transfer from DSpace to OCLC to Improve ETD Work Flow." *New Library World* 110 (5–6): 249–64. https://doi.org/10.1108/03074800910954271.

"Embedding Projector - Visualization of High-Dimensional Data." n.d. Accessed July 30, 2020. https://projector.tensorflow.org/.

Feng, He, and Ding Xiaoqing. 2007. "Improving Naive Bayes Text Classifier Using Smoothing Methods." *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4425 LNCS: 703–7.

"François Chollet - Google Scholar Citations." n.d. Accessed May 11, 2020. https://scholar.google.com/citations?user=VfYhf2wAAAAJ&hl=en.

Garcia, Edel. 2016. "A Tutorial on the BM25F Model," no. October: 1–9.

Golub, Koraljka, Thierry Hamon, and Anders Ardö. 2007. "Automated Classification of Textual Documents Based on a Controlled Vocabulary in Engineering." *Knowledge Organization* 34 (4): 247–63. https://doi.org/10.5771/0943-7444-2007-4-247.

Grzegorczyk, Karol. 2019. "Vector Representations of Text Data in Deep Learning." http://arxiv.org/abs/1901.01695.

Han, Hui, C. L. Giles, E. Manavoglu, Hongyuan Zha, Zhenyue Zhang, and E. A. Fox. 2003. "Automatic Document Metadata Extraction Using Support Vector Machines." *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries* 2003-Janua (May 2014): 37–48. https://doi.org/10.1109/JCDL.2003.1204842.

"Home Page for 20 Newsgroups Data Set." n.d. Accessed May 7, 2020a. http://qwone.com/~jason/20Newsgroups/.

Ikonomakis, M., Sotos Kotsiantis, and V. Tampakas. 2005. "Text Classification Using Machine Learning Techniques." *WSEAS Transactions on Computers* 4 (8): 966–74. https://doi.org/10.11499/sicejl1962.38.456.

"ISO - ISO 15836-1:2017 - Information and Documentation — The Dublin Core Metadata Element Set — Part 1: Core Elements." n.d. Accessed May 7, 2020. https://www.iso.org/standard/71339.html.

Iyyer, Mohit, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé. 2015. "Deep Unordered Composition Rivals Syntactic Methods for Text Classification." *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference* 1: 1681–91. https://doi.org/10.3115/v1/p15-1162.

Jang, Beakcheol, Inhwan Kim, and Jong Wook Kim. 2019. "Word2vec Convolutional Neural Networks for Classification of News Articles and Tweets." *PLoS ONE* 14 (8): 1–20. https://doi.org/10.1371/journal.pone.0220976.

Jayakanth, F., K. Maly, M. Zubair, and L. Aswath. 2005. "Approaches to Make CDS/ISIS Databases Interoperable with OAI-Compliant Digital Libraries." *Program* 39 (3): 269–78. https://doi.org/10.1108/00330330510610609.

Joseph, Sethunya R, Hlomani Hloman, Keletso Letsholo, and Kutlwano Sedimo. 2016. "Natural Language Processing: A Review." *International Journal of Research in Engineering and Applied Sciences* 6 (3): 1–8.

Kennedy, Algorithms, and Odhiambo Ogada. 2016. "N-Grams for Text Classification Using Supervised Machine Learning."

"Keras: The Python Deep Learning API." n.d. Accessed May 8, 2020. https://keras.io/.

Kerstin Denecke, Thomas Risse. 2009. "30 Text Classification Based on Limited Bibliographic Metadata∗.Pdf." In *Text Classification Based on Limited Bibliographic Metadata*.

Kim, Yoon. 2014. "Convolutional Neural Networks for Sentence Classification." *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 1746–51. https://doi.org/10.3115/v1/d14-1181.

Kocmi, Tom, and Ondřej Bojar. 2017. "An Exploration of Word Embedding Initialization in Deep-Learning Tasks." http://arxiv.org/abs/1711.09160.

Kowsari, Kamran, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. 2019. "Text Classification Algorithms: A Survey." *Information (Switzerland)*. https://doi.org/10.3390/info10040150.

Kushwaha, Vijendra. 2015. "Harvesting Metadata from Open Educational Resources for Semantic Annotation of Online Educational Content," 1–10.

Le, Quoc V., and Tomas Mikolov. 2014. "Distributed Representations of Sentences and Documents." *31st International Conference on Machine Learning, ICML 2014* 4 (May): 2931–39. http://arxiv.org/abs/1405.4053.

Lilleberg, Joseph, Yun Zhu, and Yanqing Zhang. 2015. "Support Vector Machines and

Word2vec for Text Classification with Semantic Features." *Proceedings of 2015 IEEE 14th International Conference on Cognitive Informatics and Cognitive Computing, ICCI*CC 2015*, 136–40. https://doi.org/10.1109/ICCI-CC.2015.7259377.

Manral, Jai, and Mohammed Alamgir Hossain. 2012. "An Innovative Approach for Online Meta Search Engine Optimization," 1–7.

Marinai, Simone. 2009. "Metadata Extraction from PDF Papers for Digital Library Ingest." *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 251–55. https://doi.org/10.1109/ICDAR.2009.232.

Martinez-Gil, Jorge. 2014. "An Overview of Textual Semantic Similarity Measures Based on Web Intelligence." *Artificial Intelligence Review* 42 (4): 935–43. https://doi.org/10.1007/s10462-012-9349-8.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. "Efficient Estimation of Word Representations in Vector Space." *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 1–12.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. "Distributed Representations Ofwords and Phrases and Their Compositionality." *Advances in Neural Information Processing Systems*, 1–9.

Naili, Marwa, Anja Habacha Chaibi, and Henda Hajjami Ben Ghezala. 2017. "Comparative Study of Word Embedding Methods in Topic Segmentation." *Procedia Computer Science* 112: 340–49. https://doi.org/10.1016/j.procs.2017.08.009.

Náther, Peter. 2005. "(Thesis) N-Gram Based Text Categorization."

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. 2014. "GloVe: Global Vectors for Word Representation." *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 1532–43. https://doi.org/10.3115/v1/d14-1162.

Ramos, Juan. 2003. "Using TF-IDF to Determine Word Relevance in Document Queries."

Richter, Georg, and Andrew MacFarlane. 2005. "The Impact of Metadata on the Accuracy of Automated Patent Classification." *World Patent Information* 27 (1): 13–26. https://doi.org/10.1016/j.wpi.2004.08.001.

Riley, Jenn (NISO). 2017. *Understanding Metadata - What Is Metadata?*

Robertson, Stephen, and Hugo Zaragoza. 2009. *The Probabilistic Relevance Framework: BM25 and Beyond. Foundations and Trends in Information Retrieval*. Vol. 3. https://doi.org/10.1561/1500000019.

Roy, Samrat Guha, B Sutradhar, and Partha Pratim Das. 2017. "Large-Scale Metadata Harvesting—Tools, Techniques and Challenges: A Case Study of National Digital Library (NDL)." *World Digital Libraries: An International Journal* 10 (1). https://doi.org/10.18329/09757597/2017/10101.

Sebastiani, Fabrizio. 2002. “Machine Learning in Automated Text Categorization.” *ACM Computing Surveys* 34 (1): 1–47. https://doi.org/10.1145/505282.505283.

“TensorFlow.” n.d. Accessed May 8, 2020. https://www.tensorflow.org/.

“The Stanford Natural Language Processing Group.” 2015. https://nlp.stanford.edu/.

Ting, S. L., W. H. Ip, and Albert H.C. Tsang. 2011. “Is Naïve Bayes a Good Classifier for Document Classification?” *International Journal of Software Engineering and Its Applications* 5 (3): 37–46.

Ulli Waltinger1, Alexander Mehler2, Mathias L¨osch, and Wolfram Horstmann. n.d. “53 Hierarchical Classification of OAI Metadata Using the DDC Taxonomy.Pdf.” Germany.

“Universal-Sentence-Encoder | TensorFlow Hub.” n.d. Accessed May 11, 2020. https://tfhub.dev/google/universal-sentence-encoder/1.

Violos, John, Konstantinos Tserpes, Iraklis Varlamis, and Theodora Varvarigou. 2018. “Text Classification Using the N-Gram Graph Representation Model Over High Frequency Data Streams.” *Frontiers in Applied Mathematics and Statistics* 4 (September): 1–19. https://doi.org/10.3389/fams.2018.00041.

Waltman, Ludo, and Nees Jan Van Eck. 2012. “A New Methodology for Constructing a Publication-Level Classification System of Science.” *Journal of the American Society for Information Science and Technology* 63 (12): 2378–92. https://doi.org/10.1002/asi.22748.

Wang, Jun. 2009. “An Extensive Study on Automated Dewey Decimal Classification.” *Journal of the American Society for Information Science and Technology* 60 (11): 2269–86. https://doi.org/10.1002/asi.21147.

Weber, Tobias, Dieter Kranzlmüller, Michael Fromm, and Nelson Tavares de Sousa. 2019. “Using Supervised Learning to Classify Metadata of Research Data by Discipline of Research,” 0–26. http://arxiv.org/abs/1910.09313.

Weber, Tobias, Dieter Kranzlmüller, Michael Fromm, and Nelson Tavares De Sousa. 2019. “Using Supervised Learning to Classify Metadata of Research Data by Discipline of Research.” https://datacite.org.

Zhang, Xiang, Junbo Zhao, and Yann Lecun. 2015. “Character-Level Convolutional Networks for Text Classification.” *Advances in Neural Information Processing Systems* 2015-Janua: 649–57.

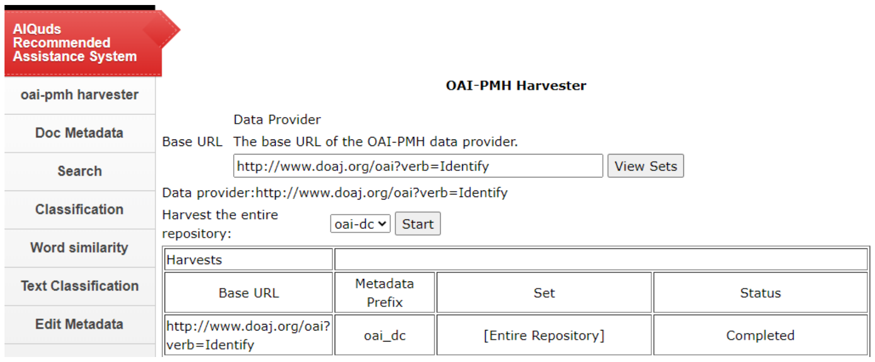
- ## List of Appendices

1. Scenarios of Al-Quds System usage

This section aims to demonstrate how Al-Quds System model works for targeted users, such as digital repositories, authors, search engines, and searchers.

1.1 Digital Repertories

The digital repertories admin can harvest and classify documents by using the Harvester model and Classification model. The harvester uses OAI-PMH protocol as shown in Fig (1) to harvest publications from other open-access digital repertories. Another capability of Al-Quds System can exploit by digital repositories is the classification as shown in Fig (2), where the classification model uses CNN or MNB classifiers.

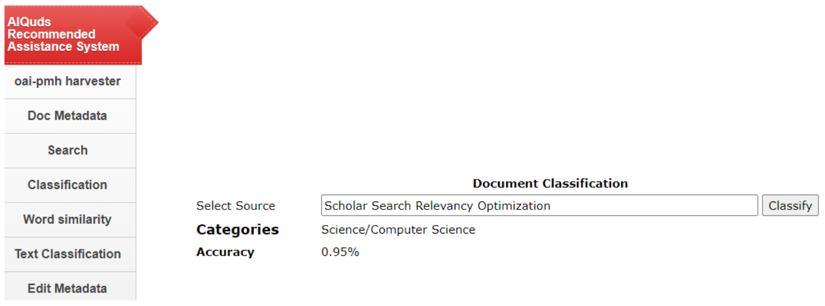


OAI-PMH Harvester

Data Provider
Base URL The base URL of the OAI-PMH data provider.
http://www.doaj.org/oai?verb=Identify
Data provider:http://www.doaj.org/oai?verb=Identify
Harvest the entire repository: oai-dc

Harvests			
Base URL	Metadata Prefix	Set	Status
http://www.doaj.org/oai?verb=Identify	oai_dc	[Entire Repository]	Completed

Figure 1: The Harvester Model



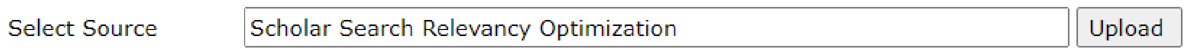
Document Classification

Select Source	Scholar Search Relevancy Optimization	Classify
Categories	Science/Computer Science	
Accuracy	0.95%	

Figure 2: The Classification Model

1.2 Authors and Indexers

Once the harvesting process complete, indexer or author need to define the main keywords for each specific document. For demonstration purposes, the (Scholar Search Relevancy optimization) document will used as shown in Fig (3).



Select Source	Scholar Search Relevancy Optimization	Upload

Figure 3: Document upload

- The highest-ranking words extract from document title and abstract, to find the surrounding content words using the TF-IDF algorithm, as shown in Fig (4).

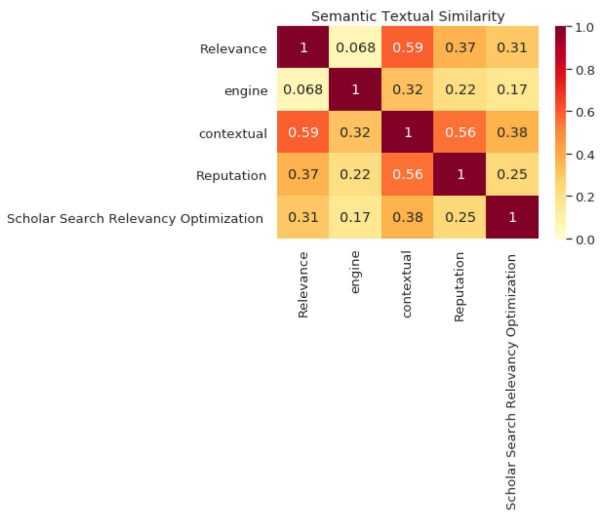
```
                TF-IDF
scholar       0.534046
relevancy     0.534046
optimization  0.534046
search        0.379978
accuracy      0.000000
publications  0.000000
possible      0.000000
primarily     0.000000
processing    0.000000
propose       0.000000
```

Figure 4: Top 10 ranking words

1.3 Search Engine

After extract the highest-ranking words, the author uses the similarity check model to find the semantic similarity words for each of the highest-ranking words, and check the relevancy score between these words and publication title. For demonstration purposes, we use two of the highest-ranking words, as shown in Fig (5).

“Relevance” Keyword



“Optimizing” Keyword

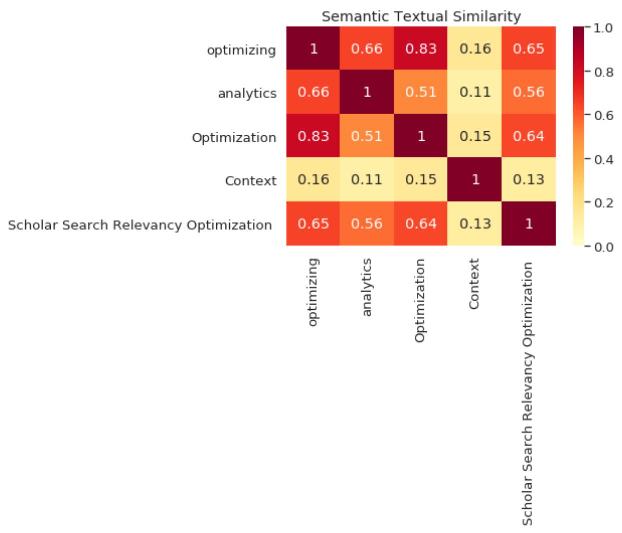
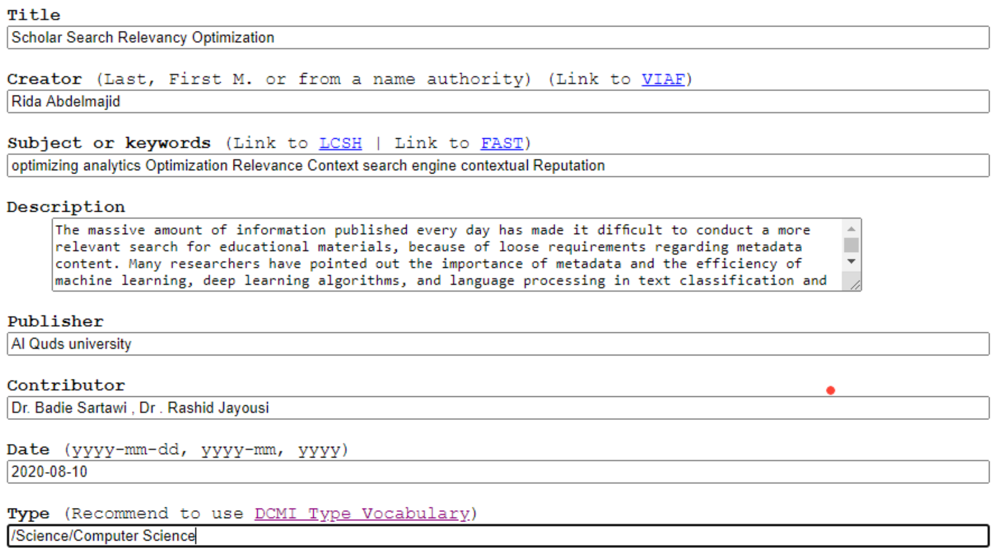


Figure 5: Similarity Check Model

Based on extracted keywords and their relevance words, author update document metadata to allow the search engines to crawl and index, in other words, Based on extracted keywords and their relevant words, author update document metadata to allow the search engine to crawl and index, as shown in Fig (6).



Title
Scholar Search Relevancy Optimization

Creator (Last, First M. or from a name authority) (Link to VIAF)
Rida Abdelmajid

Subject or keywords (Link to LCSH | Link to FAST)
optimizing analytics Optimization Relevance Context search engine contextual Reputation

Description
The massive amount of information published every day has made it difficult to conduct a more relevant search for educational materials, because of loose requirements regarding metadata content. Many researchers have pointed out the importance of metadata and the efficiency of machine learning, deep learning algorithms, and language processing in text classification and

Publisher
Al Quds university

Contributor
Dr. Badie Sartawi , Dr . Rashid Jayousi

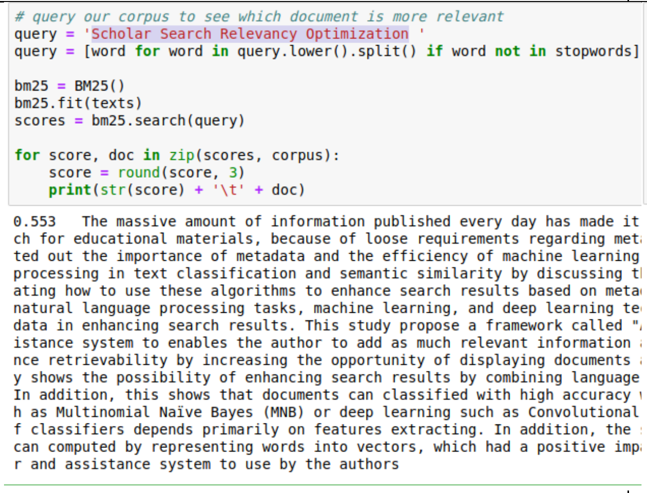
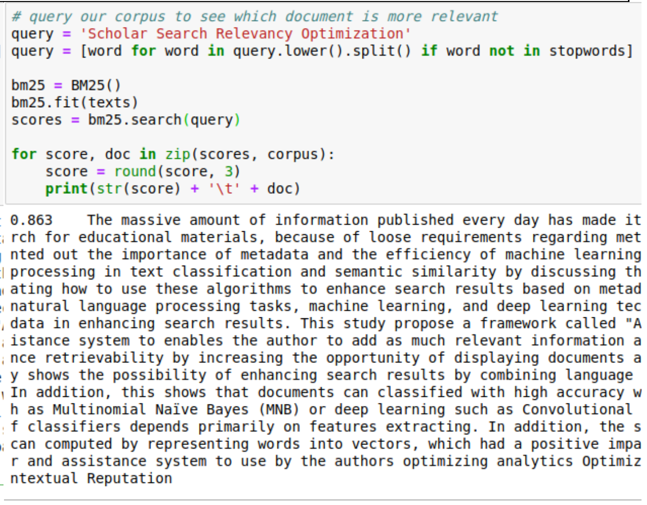
Date (yyyy-mm-dd, yyyy-mm, yyyy)
2020-08-10

Type (Recommend to use DCMI Type Vocabulary)
/Science/Computer Science

Figure 6: Metadata Model

The final step is to check the ranking score of search result relevance, and presentation. Authors can compute the relevance score between search terms and display results, for demonstration purpose, we used two different search query. Table (1) shown a comparison score result by using Al-Quds System score results, and without using Al-Quds System.

Table 1: Ranking Score Results

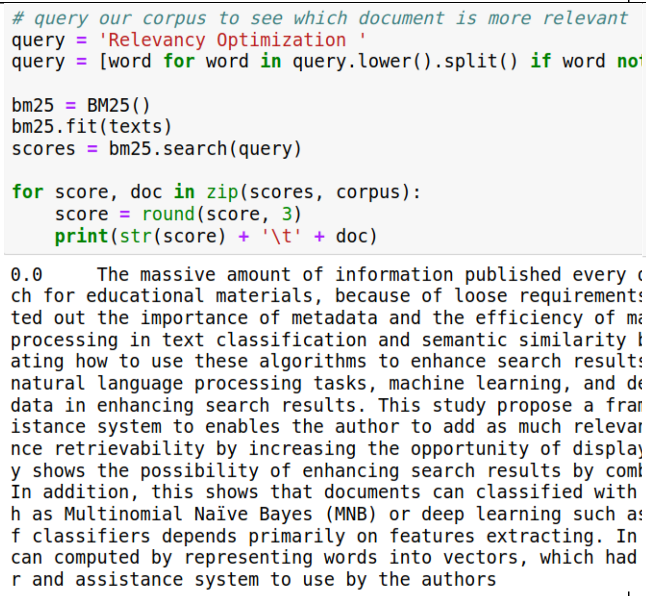
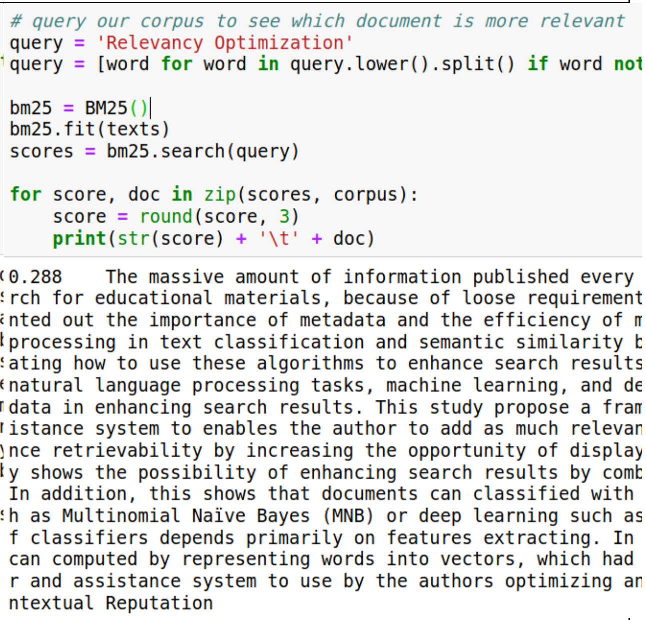
Ranking Score without Al-Quds System	Ranking Score with Al-Quds System
	
Ranking Score: .553	Ranking Score: .863

```
# query our corpus to see which document is more relevant
query = 'Scholar Search Relevancy Optimization '
query = [word for word in query.lower().split() if word not in stopwords]

bm25 = BM25()
bm25.fit(texts)
scores = bm25.search(query)

for score, doc in zip(scores, corpus):
    score = round(score, 3)
    print(str(score) + '\t' + doc)
```

```
0.553   The massive amount of information published every day has made it
ch for educational materials, because of loose requirements regarding meta
ted out the importance of metadata and the efficiency of machine learning
processing in text classification and semantic similarity by discussing th
ating how to use these algorithms to enhance search results based on meta
natural language processing tasks, machine learning, and deep learning te
data in enhancing search results. This study propose a framework called "
istance system to enables the author to add as much relevant information
nce retrievability by increasing the opportunity of displaying documents
y shows the possibility of enhancing search results by combining language
In addition, this shows that documents can classified with high accuracy
h as Multinomial Naïve Bayes (MNB) or deep learning such as Convolutional
f classifiers depends primarily on features extracting. In addition, the
can computed by representing words into vectors, which had a positive imp
r and assistance system to use by the authors
```

```
# query our corpus to see which document is more relevant
query = 'Scholar Search Relevancy Optimization'
query = [word for word in query.lower().split() if word not in stopwords]

bm25 = BM25()
bm25.fit(texts)
scores = bm25.search(query)

for score, doc in zip(scores, corpus):
    score = round(score, 3)
    print(str(score) + '\t' + doc)
```

```
0.863    The massive amount of information published every day has made it
rch for educational materials, because of loose requirements regarding met
nted out the importance of metadata and the efficiency of machine learning
processing in text classification and semantic similarity by discussing th
ating how to use these algorithms to enhance search results based on metad
natural language processing tasks, machine learning, and deep learning tec
data in enhancing search results. This study propose a framework called "A
istance system to enables the author to add as much relevant information a
nce retrievability by increasing the opportunity of displaying documents a
y shows the possibility of enhancing search results by combining language
In addition, this shows that documents can classified with high accuracy w
h as Multinomial Naïve Bayes (MNB) or deep learning such as Convolutional
f classifiers depends primarily on features extracting. In addition, the s
can computed by representing words into vectors, which had a positive impa
r and assistance system to use by the authors optimizing analytics Optimiz
ntextual Reputation
```

	
Ranking Score: 0	Ranking Score: .288

```
# query our corpus to see which document is more relevant
query = 'Relevancy Optimization '
query = [word for word in query.lower().split() if word no

bm25 = BM25()
bm25.fit(texts)
scores = bm25.search(query)

for score, doc in zip(scores, corpus):
    score = round(score, 3)
    print(str(score) + '\t' + doc)
```

```
0.0     The massive amount of information published every
ch for educational materials, because of loose requirement
ted out the importance of metadata and the efficiency of m
processing in text classification and semantic similarity
ating how to use these algorithms to enhance search result
natural language processing tasks, machine learning, and d
data in enhancing search results. This study propose a fra
istance system to enables the author to add as much releva
nce retrievability by increasing the opportunity of displa
y shows the possibility of enhancing search results by com
In addition, this shows that documents can classified with
h as Multinomial Naïve Bayes (MNB) or deep learning such a
f classifiers depends primarily on features extracting. In
can computed by representing words into vectors, which had
r and assistance system to use by the authors
```

```
# query our corpus to see which document is more relevant
query = 'Relevancy Optimization'
query = [word for word in query.lower().split() if word not

bm25 = BM25()
bm25.fit(texts)
scores = bm25.search(query)

for score, doc in zip(scores, corpus):
    score = round(score, 3)
    print(str(score) + '\t' + doc)
```

```
0.288   The massive amount of information published every
rch for educational materials, because of loose requirement
nted out the importance of metadata and the efficiency of m
processing in text classification and semantic similarity b
ating how to use these algorithms to enhance search results
natural language processing tasks, machine learning, and de
data in enhancing search results. This study propose a fram
istance system to enables the author to add as much relevan
nce retrievability by increasing the opportunity of display
y shows the possibility of enhancing search results by comb
In addition, this shows that documents can classified with
h as Multinomial Naïve Bayes (MNB) or deep learning such as
f classifiers depends primarily on features extracting. In
can computed by representing words into vectors, which had
r and assistance system to use by the authors optimizing an
ntextual Reputation
```

2. Requirement

2.1 Python 2.7 or Above
2.2 Jupiter Notebook
2.3 Python Modules
2.4 Scikit-Learn
2.5 Numpy
2.6 Matplotlib

3. GitHub Project

https://github.com/rabdelmajid/Master-Thesis-.git

4. Model Source Code

The source code can be viewed online, via the URL given below, and it can be downloaded via the URL given in the section 3 Program download

4.1 Multinomial Naive Bayes

4.2	Naive Bayes classification on 20newsgroup dataset-TF-IDF
4.3	sentence encoder similarity check
4.4	CNN_ Adam
4.5	CNN_ rmsprop
4.6	CNN_ sgd
4.7	CNN 10 Epoch
4.8	CNN 15 epoch
4.9	BM25
4.10	TF-IDF

5. Model Download

The download is provided as a Python package, which includes the source code of the program.

https://github.com/rabdelmajid/Master-Thesis-

6. Model Documentation

The program documentation can be viewed online, or downloaded within the source code.

https://github.com/rabdelmajid/Master-Thesis-.git

7. Prototype Html Files

https://github.com/rabdelmajid/Master-Thesis-/blob/master/Prototype.rar